Deep Generative Models for Reject Inference in Credit Scoring

Rogelio A. Mancisidor[a,b,*] rogelio.a.mancisidor@uit.no
Michael Kampffmeyer [a] michael.c.kampffmeyer@uit.no
Kjersti Aas [c] kjersti@nr.no
Robert Jenssen [a] robert.jenssen@uit.no

[a]Machine Learning Group, Department of Physics and Technology, Faculty of Science and Technology, UiT - The Artic University of Norway, Hansine Hansens veg 18, Tromsø 9037, Norway
[b]Credit Risk Models, Santander Consumer Bank AS, Strandveien 18, Lysaker 1325, Norway
[c]Statistical Analysis, Machine Learning and Image Analysis
Norwegian Computing Center, Gaustadalleen 23a, Oslo 0373, Norway

[*]Corresponding author

February 3, 2020

Abstract. Credit scoring models based on accepted applications may be biased and their consequences can have a statistical and economic impact. Reject inference is the process of attempting to infer the creditworthiness status of the rejected applications. Inspired by the promising results of semi-supervised deep generative models, this research develops two novel Bayesian models for reject inference in credit scoring combining Gaussian mixtures and auxiliary variables in a semi-supervised framework with generative models. To the best of our knowledge this is the first study coupling these concepts together. The goal is to improve the classification accuracy in credit scoring models by adding reject applications. Further, our proposed models infer the unknown creditworthiness of the rejected applications by exact enumeration of the two possible outcomes of the loan (default or non-default). The efficient stochastic gradient optimization technique used in deep generative models makes our models suitable for large data sets. Finally, the experiments in this research show that our proposed models perform better than classical and alternative machine learning models for reject inference in credit scoring, and that model performance increases with the amount of data used for model training.

Keywords: Reject Inference, Deep Generative Models, Credit Scoring, Semi-Supervised Learning

1 Introduction

Credit scoring uses statistical models to transform the customers' data into a measure of the borrowers' ability to repay the loan [1]. These models are developed, commonly, based on accepted applications because the bank knows whether the customer repaid the loan. The problem is that this data sample is biased since it excludes the rejected applications systematically. This is called selection bias.

Using a biased sample to estimate any model has several problems. The straightforward consequence is

that the model parameters are biased [2], which has a statistical and economic impact [3, 4]. Another consequence is that the default probability can be underestimated, affecting the risk premium and the profitability of the bank [5]. Hence, reject inference, which is the process of attempting to infer the true creditworthiness status of the rejected applications [6], has created a great deal of interest.

There is a vast literature on reject inference using classical statistical methods. However, there has been little research using machine learning techniques (see Table 1). Semi-supervised learning approaches design and train models using labeled (accepted applications) and unlabeled data (rejected applications), and aim to utilize the information embedded in both data to improve the classification of unseen observations. There are several fields where semi-supervised deep generative models have achieved state-of-the-art results, e.g. in semi-supervised image classification [7, 8], in semi-supervised sentiment analysis [9, 10], and in unsupervised clustering [11]. Additionally, the useful information embedded in their latent space is well documented [12, 13, 14, 15]. Inspired by the modeling framework introduced by [7], this research develops two novel models for reject inference models in credit scoring combining, for the first time, auxiliary variables [8] and Gaussian mixtures parametrized by neural networks in a semi-supervised framework.

Our proposed models have a flexible latent space, induced by the Gaussian mixtures, to improve the variational approximation and the reconstruction of the input data [8, 16]. In addition, one of our models not only uses the input data to classify new loan applications, but also a latent representation of it. This makes the classifier more expressive [8, 16]. We compare the performance of the semi-supervised generative models with a range of techniques representing the state-of-the-art in reject inference for credit scoring, including three classical reject inference techniques (reclassification, fuzzy parceling[1] and augmentation [17]), and three semi-supervised machine learning approaches (self-learning [18] MLP, self-learning SVM, and semi-supervised SVM [19]). Additionally, we include two supervised machine learning models (multilayer perceptron (MLP) [20] and support vector machine (SVM) [21]) to measure the marginal gain of reject inference.

To summarize, the main contributions of this paper are as follows:

1. We develop two novel reject inference models for credit scoring combining auxiliary variables and Gaussian mixtures in a semi-supervised framework with generative models for the first time.
2. We derive the objective functions for our proposed models and show how they can be parameterized by MLPs and optimized with stochastic gradient descent.
3. We parametrize the Gaussian mixtures using an MLP and we show how to train them with semi-supervised data.
4. Our empirical results show that our proposed models achieve higher performance compared to the state-of-art methods in credit scoring. Additionally, the model performance for our proposed models increases with the amount of data used for training.

The rest of the paper is organized as follows. Section 2 reviews the related work on reject inference in credit risk, then Section 3 presents an overview of semi-supervised deep generative models and introduces the proposed models. Section 4 explains the data, methodology and main results. Finally, Section 5 presents the main conclusion of this research.

2 Related Work

Banks decide whether to grant credit to new applications as well as how to deal with existing customers, e.g. deciding whether credit limits should be increased and determining which marketing campaign is most appropriate. The tools that help banks with the first problem are called credit scoring models, while behavioral scoring models are used to handle exiting customers [22]. Both type of models estimate the ability that a borrower will be unable to meet its debt obligations, which is referred to as default probability. This research focuses on reject inference to improve the classification accuracy of credit

[1] For a review of the reclassification and fuzzy parceling approaches see [1, 3].

(Year) Author	Data type	Status of rejects	No. of accepts	No. of rejects	Reject Inference approach	Classification method
(1993) Joanes [25]	Artificial	Unknown	75	12	Reclassification	Logistic
(2000) Feelders [24]	Artificial	Unknown	Varying	Varying	EM	QDA, Logistic
(2001) Chen and Åstebro [4]	Coorporate	Known	298	599	Heckman's model	Probit, Bivariate probit
(2003) Banasik et al. [26]	Consumer	Known	8 168	4 040	Augmentation	Logistic, Probit
(2004) Crook and Banasik [27]	Consumer	Known	8 168	4 040	Augmentation, Extrapolation	Logistic
(2004) Verstraeten and Van den Poel [28]	Consumer	Partially known	38 048	6 306	Augmentation	Logistic
(2005) Banasik and Crook [29]	Consumer	Known	8 168	4 040	Augmentation	Logistic
(2006) Sohn and Shin [30]*	Consumer	Unknown	759	10	Reclassication	Survival analysis
(2007) Banasik and Crook [31]	Consumer	Known	8 168	4 040	Augmentation and Heckman's model	Logistic, Bivariate probit
(2007) Kim and Sohn [32]	Corporate	Known	4 298	689	Heckman's model	Bivariate probit
(2007) Wu and Hand [33]	Artificial	Known	Varying	Varying	Heckman's model	OLS, Bivariate Probit
(2010) Banasik and Crook [34]*	Consumer	Known	147 179	Varying	Augmentation	Survival analysis
(2010) Marshall et al. [5]	Consumer	Known	40 700	2 934	Heckman's model	Probit, Bivariate probit
(2010) Maldonado and Paredes [35]	Consumer	Known	800	200	Extrapolation	SVM
(2012) Chen and Åstebro [36]	Corporate	Known	4 589	Varying	Bound and Collapse	Bayesian
(2013) Bücker et al. [2]	Consumer	Unknown	3 984	5 667	Augmentation	Logistic
(2013) Anderson and Hardin [37]	Consumer	Unknown	3 000	1 500	Augmentation, EM	Logistic
(2016) Nguyen [3]	Consumer	Unknown	56 016	142 571	Augmentation, Extrapolation	Logistic
(2017) Li et al. [23]	Consumer	Unknown	56 626	563 215	Extrapolation	Semi-supervised SVM

Table 1: Up to date research overview on reject inference. The scope of the research marked with * differs from ours, hence they are included in Section 2.

scoring models by utilizing the rejected applications. In Table (1), we present an updated research overview on reject inference in credit scoring extending the one presented in [23].

There are two broad approaches to estimate the default probability; the function estimation model (e.g. logistic regression) and the density estimation approach (e.g. linear discriminant analysis). The latter is more susceptible to provide biased parameter estimates when the rejected applications are ignored [6, 24].

According to [6], reject inference represents several challenges. First of all, when attempting to correct the selection bias, the customer characteristics used to develop the current credit scoring model must be available. Otherwise, including the rejected applications in the new model might be insufficient to correct the selection bias. Some techniques, such as mixture decomposition, require assumptions about the default and non-default distributions. In general, these distributions are unknown. Finally, the methods based on supplementary credit information about the reject applications, which might be bought at credit bureaus, can be unrealistic for some financial institutions. Either they cannot afford to pay for it or the data may not be available.

A simple approach for reject inference is augmentation [17]. In this approach, the accepted applications are re-weighted to represent the entire population. The common way to find these weights is using the accept/reject probability. For example if a given application has a probability of being rejected of 0.80, then all similar applications would be weighted up $1/(1-0.8) = 5$ times [1]. None of the empirical research using augmentation shows significant improvements in either correcting the selection bias or improving model performance, see [1, 2, 26, 27, 28, 29, 31]. The augmentation technique assumes that the default probability is independent of whether the loan is accepted or rejected [38]. However, [32] shows empirically that this assumption is wrong.

Heckman's bivariate two-stage model [39, 40] has been used in different reject inference studies[2]. This approach simultaneously models the accept/reject and default/non-default mechanisms. Assuming that the error terms in these processes are bivariate normally distributed with unit variance and correlation coefficient ρ, the selection bias arises when $\rho \neq 0$ and it is corrected using the inverse of the Mills ratio.

Despite the popularity of Heckman's model, it is unclear whether this model can correct the selection bias or improve model performance. Some studies claim either higher model performance or different model parameters after using Heckman's model [5, 26, 31, 32, 42]. These results, as explained by [4], depend upon whether the selection and default equations are correlated. On the other hand, [33, 36, 43] state that the model parameters are inefficient, and the main criticism is that the Heckman's model fails to correct the selection bias when it is strong. This happens either when the correlation between the error terms in the selection and outcome equations is high or the data has high degree of censoring [43].

A comparison of different reject inference methods, e.g. augmentation, parceling, fuzzy parceling and the

[2]The Heckman's model, named after Nobel Laureate James Joseph Heckman, has been extended or modified in different directions. See [4] for a chronological overview of the model evolution and its early applications. It was in [41] where the Heckman's approach was first applied to credit scoring where the outcome is discrete.

Heckman's model, is presented in [3]. The parceling and fuzzy parceling methods are very similar. They first fit a logistic regression model using the accepted applications. Then they use this model to estimate the default probability for all rejected applications. The difference is that the parceling method chooses a threshold on the default probability to assign the unknown outcome y to the rejected applications. On the other hand, the fuzzy parceling method assumes that each reject application has both outcomes $y = 1$ and $y = 0$, with weights given by the fitted model using only the accepted applications. Finally, the parcelling (fuzzy parceling) method fits a new (weighted) logistic regression using both accepted and rejected applications. The results in [3] do not show higher model performance using the reject inference methods. However, the parameter estimates are different when applying the augmentation and parceling approaches. Hence, reject inference has a statistical and economic impact on the final model in this case.

Support vector machines are used in [35] to extend the self-training (SL) algorithm, by adding the hypothesis that the rejected applications are riskier[3]. Specifically, their approach iteratively adds rejected applications with higher confidence, i.e. vectors far from the decision-hyperplane, to retrain a SVM (just as in the SL algorithm). However, vectors close to the hyperplane are penalized since the uncertainty about their true label is higher. Their proposed iterative approach shows superior performance compared to other reject inference configurations using SVMs, including semi-supervised support vector machines (S3VM). In addition to higher performance, the iterative procedure in [35] is faster than the S3VM.

The S3VM model is used in [23] for reject inference in credit scoring [4] using the accepted and rejected applications to fit an optimal hyperplane with maximum margin. The hyperplane traverses trough non-density regions of rejected applications and, at the same time, separates the accepted applications. Their results show higher performance compared to the logit and supervised support vector machine models. In Section 4, we show that S3VM does not scale to large credit scoring data sets and that our proposed models are able to use, at least, 16 times more data compared to S3VM.

In [24] Gaussian mixture models (GMM) are used for density estimation of the default probability. The idea is that each component in the mixture density models a class-conditional distribution. Then, the model parameters are estimated using the expectation-maximization (EM) algorithm, which can estimate the parameters even when the class labels for the rejected applications are missing. The EM algorithm is also used for reject inference in [37]. Both papers report high model performance. However, the results in [24] are based on artificial data and [37] only judge performance based on the Confusion matrix. Finally, the major limitation of the EM algorithm is that we need to be able to estimate the expectation over the latent variables. We show in Section 3 that deep generative models circumvent this restriction by approximation.

A Bayesian approach for reject inference is presented in [36]. In this method the default probability is inferred from the missing data mechanism. The authors use the bound-collapse approach [5] to estimate the posterior distribution over the score and class label, which is assumed to have a Dirichlet distribution as well as the marginal distribution of the missing class label. The reason for using the bound-collapse method is to avoid exhaustive numerical procedures, like the Gibbs Sampling, to estimate the posterior distributions in this model. Their results show that the Bayesian bound-collapse method perform better than the augmentation and Heckman's model.

In this research we propose a novel Bayesian inference approach for reject inference in credit scoring, which uses Gaussian mixture models and differs from [24, 36] in that our models are based on variational inference, neural networks, and stochastic gradient optimization. The main advantages of our proposed method are that (i) inference of the rejected applications is based on an approximation of the posterior distribution and on the exact enumeration of the two possible outcomes that the rejected applications could have taken, (ii) the models use a latent representation of the customers' data, which contain powerful information, and (iii) deep generative models scale to large data sets.

[3]The self-training algorithm is an iterative approach where highly confident predictions about the unlabeled data are added to retrain the model. This procedure is repeated as many times as the user specify it. The main criticism of this method is that it can strengthen poor predictions [7].

[4]The model used in [23], originally developed by [44], uses a branch-and-bound approach to solve the mixed integer constrained quadratic programming problem faced in semi-supervised SVMs. This approach reduces the training time making it suitable for large-sized problems.

[5]This model is originally presented in Sebastiani and Ramoni (2000) "Bayesian inference with missing data using bound and collapse".

3 Deep Generative Models

The principles of variational inference with deep neural networks are given in [45, 46]. Building upon this work, [7] proposed a generalized probabilistic approach for semi-supervised learning. This approach will be explained in Section 3.1 before we introduce two novel models for reject inference in credit scoring in Sections 3.2 and 3.3.

3.1 Semi-supervised Deep Generative Models for Reject Inference

In reject inference, the data set $D = \{D_{accept}, D_{reject}\}$ is composed of n (labeled) accepted applications $D_{accept} = \{(\boldsymbol{x}, y)_1, ..., (\boldsymbol{x}, y)_n\}$ and m (unlabeled) rejected applications $D_{reject} = \{\mathbf{x}_{n+1}, ..., \mathbf{x}_{n+m}\}$, where $\mathbf{x} \in \mathbb{R}^{\ell_x}$ is the feature vector and $y_i \in \{0, 1\}$ is the class label or the outcome of the loan, $y = 0$ if the customer repaid the loan, otherwise $y = 1$. Additionally, generative models assume that latent variable $\boldsymbol{z} \in \mathbb{R}^{\ell_z}$ governs the distribution of $\boldsymbol{x}$.

The goal of the generative model is to obtain the joint distribution $p(\boldsymbol{x}, y)$ of the data used for credit scoring and the outcome of the loan. However, this distribution is intractable since it requires integration over the whole latent space, i.e. $\int p(\boldsymbol{x}, y, \boldsymbol{z})d\boldsymbol{z}$. Further, the intractability of $p(\boldsymbol{x}, y)$ translates into an intractable posterior distribution of $\boldsymbol{z}$ through the relationship

$$p(\boldsymbol{z}|\boldsymbol{x}, y) = \frac{p(\boldsymbol{x}, y, \boldsymbol{z})}{\int p(\boldsymbol{x}, y, \boldsymbol{z})d\boldsymbol{z}}. \tag{1}$$

Hence, we approximate the true posterior $p(\boldsymbol{z}|\boldsymbol{x}, y)$ with the inference model $q(\boldsymbol{z}|\boldsymbol{x}, y)$ and minimize the Kullback-Leibler (KL) divergence[6] $KL[q(\boldsymbol{z}|\boldsymbol{x}, y)||p(\boldsymbol{z}|\boldsymbol{x}, y)]$ to make the approximation as close as possible to the true density.

The $KL[q(\boldsymbol{z}|\boldsymbol{x}, y)||p(\boldsymbol{z}|\boldsymbol{x}, y)]$ term, the objective function $\mathcal{L}_{accept}$, and the density $p(x, y)$ are related by the following expression

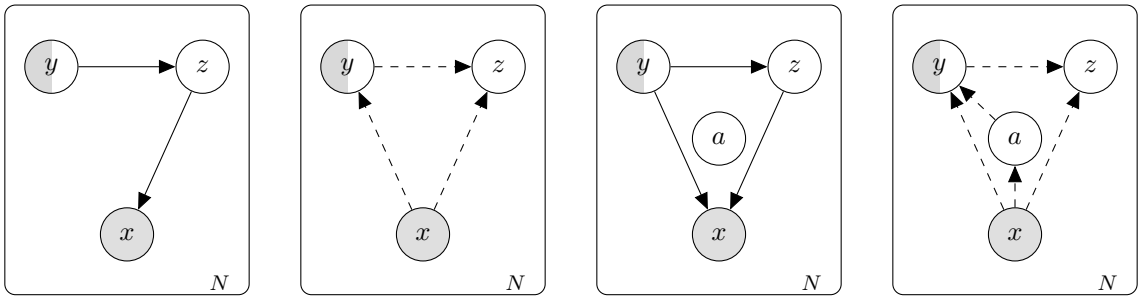
$$\begin{aligned}
\log p(\boldsymbol{x}, y) &= \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x},y)}[\log p(\boldsymbol{x}, y)] \\
&= \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x},y)}\Big[\log \frac{p(\boldsymbol{x}, y, \boldsymbol{z})}{p(\boldsymbol{z}|\boldsymbol{x}, y)} \frac{q(\boldsymbol{z}|\boldsymbol{x}, y)}{q(\boldsymbol{z}|\boldsymbol{x}, y)}\Big] \\
&= \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x},y)}\Big[\log \frac{p(\boldsymbol{x}, y, \boldsymbol{z})}{q(\boldsymbol{z}|\boldsymbol{x}, y)}\Big] + \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x},y)}\Big[\log \frac{q(\boldsymbol{z}|\boldsymbol{x}, y)}{p(\boldsymbol{z}|\boldsymbol{x}, y)}\Big] \\
&:= -\mathcal{L}_{accept}(\boldsymbol{x}, y) + KL[q(\boldsymbol{z}|\boldsymbol{x}, y)||p(\boldsymbol{z}|\boldsymbol{x}, y)].
\end{aligned} \tag{2}$$

Given that the KL divergence in Equation 2 is strictly positive, the term $-\mathcal{L}_{accept}(\boldsymbol{x}, y)$ is a lower bound on $\log p(\boldsymbol{x}, y)$, i.e. $\log p(\boldsymbol{x}, y) \geq -\mathcal{L}_{accept}(\boldsymbol{x}, y)$. Hence, since we cannot evaluate $p(\boldsymbol{z}|\boldsymbol{x}, y)$, we maximize $\log p(\boldsymbol{x}, y)$ by maximizing the negative lower bound.

Note that in Equation 2 we assume that the outcome y of the loan is known. However, this is not the case for the rejected applications D_{reject}. In this case, generative models treat y as a latent variable and approximate the true posterior distribution $p(y|\boldsymbol{x})$ with the parametric function $q(y|\boldsymbol{x})$. Assuming the factorization $q(\boldsymbol{z}, y|\boldsymbol{x}) = q(y|\boldsymbol{x})q(\boldsymbol{z}|\boldsymbol{x}, y)$ and a simple form for $q(y|\boldsymbol{x})$, we can take the explicit expectation over the class label y, i.e. we handle the uncertainty about the outcome of the loan by summing over the two possible outcomes that it might have taken. Mathematically,

$$\begin{aligned}
\mathbb{E}_{q(\boldsymbol{z},y|\boldsymbol{x})}\Big[\log \frac{p(\boldsymbol{x}, y, \boldsymbol{z})}{q(\boldsymbol{z}, y|\boldsymbol{x})}\Big] &= \mathbb{E}_{q(y|\boldsymbol{x})}\mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x},y)}\Big[\log \frac{p(\boldsymbol{x}, y, \boldsymbol{z})}{q(\boldsymbol{z}, y|\boldsymbol{x})}\Big] \\
&= \mathbb{E}_{q(y|\boldsymbol{x})}[-\mathcal{L}_{accept}(\boldsymbol{x}, y) - \log q(y|\boldsymbol{x})] \\
&= \sum_y q(y|\boldsymbol{x})[-\mathcal{L}_{accept}(\boldsymbol{x}, y) - \log q(y|\boldsymbol{x})] \\
&:= -\mathcal{L}_{reject}(\boldsymbol{x}).
\end{aligned} \tag{3}$$

[6] The KL divergence is a measure of the proximity between two densities, e.g. $KL[q(\cdot)||p(\cdot)]$, and it is commonly measured in bits. It is non-negative and it is minimized when $q(\cdot) = p(\cdot)$.



(a) Model 1: Generative (b) Model 1: Inference (c) Model 2: Generative (d) Model 2: Inference

Figure 1: Plate notation for Model 1 and Model 2 where $\boldsymbol{x}$ is the observed feature vector, y is the outcome of the loan and it is only observed for the accepted applications, and $\boldsymbol{z}$ and $\boldsymbol{a}$ are latent variables. The generative process is specified by solid lines, while the inference process is shown with dotted lines. Note that the MLP weights $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ lie outside the plates and we omit them to do not clutter the diagrams.

Therefore, the objective function in semi-supervised deep generative models is the sum of the supervised lower bound for the accepted applications and the unsupervised lower bound for the rejected applications

$$\mathcal{L} = \mathcal{L}_{accept}(\boldsymbol{x}, y) + \mathcal{L}_{reject}(\boldsymbol{x}). \tag{4}$$

Furthermore, deep generative models parametrize the parameters of the density functions in Equation 2 and 3 by multilayer perceptron (MLP) networks. For example, if $\boldsymbol{z}|\boldsymbol{x}, y$ is multivariate Gaussian distributed with diagonal covariance matrix, we use the notation

$$p(\boldsymbol{z}|\boldsymbol{x}, y) \sim \mathcal{N}(\boldsymbol{z}|\boldsymbol{x}, y;\ \boldsymbol{\mu} = f_{\boldsymbol{\theta}}(\boldsymbol{x}, y), \boldsymbol{\sigma}^2\boldsymbol{I} = f_{\boldsymbol{\theta}}(\boldsymbol{x}, y)), \tag{5}$$

where $\boldsymbol{\mu} \in \mathbb{R}^{\ell_z}$ and $\boldsymbol{\sigma}^2 \in \mathbb{R}^{\ell_z}$, to specify that the parameters of the Gaussian distribution are parametrized by an MLP network denoted by $f(\boldsymbol{x}, y)$ with input data $\boldsymbol{x}, y$ and weights $\boldsymbol{\theta}$[7]. Hence, the optimization of the objective function is with respect to the weights in the MLP. An alternative notation is to simply use the subscript $\boldsymbol{\theta}$ in the corresponding distribution, i.e. $p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x}, y)$.

Finally, note that the EM algorithm used in [24, 37] cannot be used in this context since it requires to compute the expectation of $p(\boldsymbol{z}|\boldsymbol{x}, y)$, which it is intractable. Other variational inference techniques, like mean-field or stochastic variational inference, determine different values of $\boldsymbol{\mu}_i$ and $\boldsymbol{\sigma}_i^2$ for each data point $\boldsymbol{x}_i$, which is computationally expensive. Similarly, traditional EM algorithms need to compute an expectation w.r.t the whole data set before updating the parameters. Therefore, deep generative models use complex functions of the data $\boldsymbol{x}$ (MLP networks) to estimate the best possible values for the latent variables $\boldsymbol{z}$. This allows replacing the optimization of point-specific parameters $\boldsymbol{\mu}_i$ and $\boldsymbol{\sigma}_i^2$, with a more efficient optimization of the MLP weights $\boldsymbol{\theta}$. The latter is denoted amortized inference [48].

3.2 Model 1: Generative and inference process

In this section we build upon the work done in [7, 11] to develop a new semi-supervised model with a Gaussian mixture parameterized with MLPs. The Gaussian mixture induces a flexible latent space that improves the approximation of the lower bound [8, 16]. Hence, Model 1 assumes a generative process $p_{\boldsymbol{\theta}}(\boldsymbol{x}, y, \boldsymbol{z}) = p(y)p_{\boldsymbol{\theta}}(\boldsymbol{z}|y)p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$, where $\boldsymbol{x} \perp y|\boldsymbol{z}$, with the following probability density functions

$$\begin{aligned} p(y) &\sim \text{Bernoulli}(y; \pi), \\ p(\boldsymbol{z}|y) &\sim \mathcal{N}(\boldsymbol{z}|y = k;\ \boldsymbol{\mu}_{z_k} = f_{\boldsymbol{\theta}}(y), \boldsymbol{\sigma}_{z_k}^2\mathbf{I} = f_{\boldsymbol{\theta}}(y))\ \textit{for}\ k = 0, 1, \\ p(\boldsymbol{x}|\boldsymbol{z}) &\sim \mathcal{N}(\boldsymbol{x}|\boldsymbol{z};\ \boldsymbol{\mu}_x = f_{\boldsymbol{\theta}}(\boldsymbol{z}), \boldsymbol{\sigma}_x^2\mathbf{I} = f_{\boldsymbol{\theta}}(\boldsymbol{z})). \end{aligned} \tag{6}$$

[7]Deep generative models can also be developed with convolutional neural networks (CNNs). However, CNNs require structured data like videos, images, or time-series data. The data sets in this research are feature vectors with customer's characteristics at the application time. This kind of data does not have the grid-like structure required for training CNNs. For an application of CNNs in credit scoring the reader is referred to [47].

Here $\mathcal{N}$ denotes the Gaussian distributions and $f(\cdot)$ is a multilayer perceptron model with weights denoted by $\boldsymbol{\theta}$. Furthermore, we assume that the inference process is factorized as $q(\boldsymbol{z}, y|\boldsymbol{x}) = q(y|\boldsymbol{x})q(\boldsymbol{z}|\boldsymbol{x}, y)$, with the following probability densities

$$\begin{aligned} q(y|\boldsymbol{x}) &\sim \text{Bernoulli}(y; \pi_{y|\boldsymbol{x}} = f_{\boldsymbol{\phi}}(\boldsymbol{x})), \\ q(\boldsymbol{z}|\boldsymbol{x}, y) &\sim \mathcal{N}(\boldsymbol{z}|\boldsymbol{x}, y;\ \boldsymbol{\mu}_z = f_{\boldsymbol{\phi}}(\boldsymbol{x}, y), \boldsymbol{\sigma}_z^2\mathbf{I} = f_{\boldsymbol{\phi}}(\boldsymbol{x}, y)). \end{aligned} \tag{7}$$

Again $\mathcal{N}$ is the Gaussian distribution and $f(\cdot)$ is a multilayer perceptron model with weights denoted by $\boldsymbol{\phi}$. Note that the marginal distribution $p(\boldsymbol{z})$ in the generative process is a GMM, i.e.

$$\begin{aligned} p(\boldsymbol{z}) =& \sum_y p(y)p(\boldsymbol{z}|y) \\ =& \pi\mathcal{N}(\boldsymbol{\mu}_{z_0}, \boldsymbol{\sigma}_{z_0}^2\mathbf{I}) + (1-\pi)\mathcal{N}(\boldsymbol{\mu}_{z_1}, \boldsymbol{\sigma}_{z_1}^2\mathbf{I}), \end{aligned}$$

where $(1-\pi)$ represents the prior for the default probability. The generative and inference processes are shown in Figure 1.

In the following sections, we use $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ to distinguish the expectation and variance terms in the generative process from the ones in the inference process as well as to differentiate the MLP's weights in the generative process from the ones in the inference process. Further, we derive the lower bound for the supervised and unsupervised data under our novel approach for reject inference in credit scoring.

Labeled data: Deriving the objective function $\mathcal{L}_{accept}$

We use Equation 2 and the factorization of the generative process in Equation 6 to derive the lower bound for the accepted data set D_{accept}. Hence, expanding the terms in the lower bound we obtain

$$\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x},y)}\Big[\log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}, y, \boldsymbol{z})}{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}, y)}\Big] = \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x},y)}[\log p(y) + \log p_{\boldsymbol{\theta}}(\boldsymbol{z}|y) + \log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}) - \log q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}, y)], \tag{8}$$

and taking the expectations, see Section B.2 in the Appendix, we find the negative lower bound for a single (supervised) data point, which is

$$\begin{aligned} -\mathcal{L}_{accept}(\{\boldsymbol{x}, y\}_i; \boldsymbol{\theta}, \boldsymbol{\phi}) = \frac{1}{2}\Big[\sum_{j=1}^{\ell_z}(1 + \log\sigma_{\boldsymbol{\phi}_j}^2) - \sum_{j=1}^{\ell_z}\Big(\log\sigma_{\boldsymbol{\theta}_{j,y}}^2 + \frac{\sigma_{\boldsymbol{\phi}_j}^2}{\sigma_{\boldsymbol{\theta}_{j,y}}^2} + \frac{(\mu_{\boldsymbol{\phi}_j} - \mu_{\boldsymbol{\theta}_{j,y}})^2}{\sigma_{\boldsymbol{\theta}_{j,y}}^2}\Big)\Big] + \log\pi_i \\ + \frac{1}{L}\sum_{l=1}^{L}\log\mathcal{N}(x_i|z_{i,l}). \end{aligned} \tag{9}$$

Here ℓ_z is the dimension of $\boldsymbol{z}$, $\sigma_{\cdot_j}^2$ and $\mu_{\cdot_j}$ are the j'th element of $\boldsymbol{\sigma}_{\cdot}^2$ and $\boldsymbol{\mu}_{\cdot}$ respectively, π_i is the prior distribution over the class label y_i, and L is the number of $\boldsymbol{z}_{i,l}$ samples drawn from $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}, y)$. We use the *reparametrization trick* $\boldsymbol{z}_{i,l} = \boldsymbol{\mu}_{i_{\boldsymbol{\phi}}} + \boldsymbol{\sigma}_{i_{\boldsymbol{\phi}}} \odot \boldsymbol{\epsilon}_l$, where $\boldsymbol{\epsilon}_l \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and $\odot$ denotes an element-wise multiplication, to backpropagate through $\boldsymbol{\sigma}_{\cdot}^2$ and $\boldsymbol{\mu}_{\cdot}$. Hence, the last term in Equation 9 is $\mathcal{N}(x_i|z_{i,l} = \boldsymbol{\mu}_{i_{\boldsymbol{\phi}}} + \boldsymbol{\sigma}_{i_{\boldsymbol{\phi}}} \odot \boldsymbol{\epsilon}_l)$ and we use $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}, y)$ to sample $\boldsymbol{\mu}_{i_{\boldsymbol{\phi}}}$ and $\boldsymbol{\sigma}_{i_{\boldsymbol{\phi}}}$. Note that since y is known in this case, we only need to backpropagate through its corresponding Gaussian component in the MLP parameterizing the GMM. In other words, if $y_i = 0$ the stochastic gradient optimization only updates all weights in $\boldsymbol{\mu}_{\boldsymbol{\theta}_y}$ and $\boldsymbol{\sigma}_{\boldsymbol{\theta}_y}^2$ for the first component in Figure 2. This is specified by the subscript y in Equation 9.

Unlabeled data: Deriving the objective function $\mathcal{L}_{reject}$

In this case, we treat the unknown labels y as latent variables and we approximate the true posterior distribution with $q(y|\boldsymbol{x})$. Given that $q(y|\boldsymbol{x}) \sim \text{Bernoulli}(\cdot)$ is a relatively easy distribution, we take the explicit expectation in the unsupervised lower bound. Following the steps in Equation 3 together with

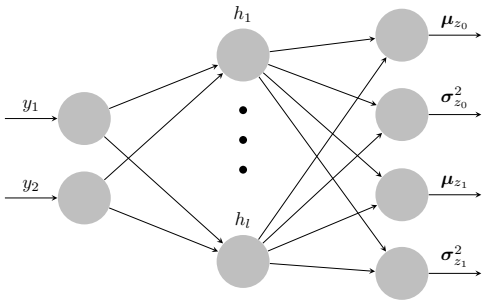


Figure 2: Gaussian mixture components parameterized by a multilayer perceptron model, where $y_\cdot$ is the one-hot-encoding for the input data ($[y_1\ y_2] = [0\ 1]$ and $[y_1\ y_2] = [1\ 0]$ are the one-hot-encoding for $y = 1$ and $y = 0$ respectively), h_l is the l'th neuron in the hidden layer, and $\boldsymbol{\mu}_{z_i}$ and $\boldsymbol{\sigma}_{z_i}$ are density moments for the i'th component in the GMM. For the accepted applications, we backpropagate trough its corresponding component, while for the rejected applications we backpropagate through both components.

the factorization in Equations 6 and 7, we obtain

$$\begin{aligned}\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z},y|\boldsymbol{x})}\Big[\log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x},y,\boldsymbol{z})}{q_{\boldsymbol{\phi}}(\boldsymbol{z},y|\boldsymbol{x})}\Big] &= \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z},y|\boldsymbol{x})}[\log p(y) + \log p_{\boldsymbol{\theta}}(\boldsymbol{z}|y) + \log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}) - \log q_{\boldsymbol{\phi}}(y|\boldsymbol{x}) \\ &\qquad - \log q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x},y)] \\ &= \mathbb{E}_{q_{\boldsymbol{\phi}}(y|\boldsymbol{x})}[-\mathcal{L}_{accept}(\boldsymbol{x};\boldsymbol{\theta},\boldsymbol{\phi}) - \log q_{\boldsymbol{\phi}}(y|\boldsymbol{x})] \\ &= \sum_y q_{\boldsymbol{\phi}}(y|\boldsymbol{x})[-\mathcal{L}_{accept}(\boldsymbol{x};\boldsymbol{\theta},\boldsymbol{\phi}) - \log q_{\boldsymbol{\phi}}(y|\boldsymbol{x})], \end{aligned} \tag{10}$$

which is, by definition, the unsupervised negative lower bound $-\mathcal{L}_{reject}(\boldsymbol{x};\boldsymbol{\theta},\boldsymbol{\phi})$. Furthermore, taking the expectations, see Section B.3 in the Appendix, we can obtain the negative lower bound for a single data point, which is

$$\begin{aligned}-\mathcal{L}_{reject}(\boldsymbol{x}_i;\boldsymbol{\theta},\boldsymbol{\phi}) = \frac{1}{2}\sum_{y=0}^{1}\pi_{y|\boldsymbol{x}_i}\Big[\sum_{j=1}^{\ell_z}(1+\log\sigma^2_{\boldsymbol{\phi}_j}) - \sum_{j=1}^{\ell_z}\Big(\log\sigma^2_{\boldsymbol{\theta}_{j,y}} + \frac{\sigma^2_{\boldsymbol{\phi}_j}}{\sigma^2_{\boldsymbol{\theta}_{j,y}}} + \frac{(\mu_{\boldsymbol{\phi}_j} - \mu_{\boldsymbol{\theta}_{j,y}})^2}{\sigma^2_{\boldsymbol{\theta}_{j,y}}}\Big)\Big] \\ + \sum_{y=0}^{1}\pi_{y|\boldsymbol{x}_i}\log\frac{\pi}{\pi_{y|\boldsymbol{x}_i}} + \frac{1}{L}\sum_{l=1}^{L}\log\mathcal{N}(\boldsymbol{x}_i|z_{i,l}),\end{aligned} \tag{11}$$

where $\pi_{y|\boldsymbol{x}}$ is the y'th element of the posterior probability over the class labels $\boldsymbol{\pi}_{y|\boldsymbol{x}} = [\pi_{y=0|\boldsymbol{x}}\ (1-\pi_{y=0|\boldsymbol{x}})]$. The rest of the parameters have the same interpretation as in the supervised negative lower bound. Note that in this case we take the expectation over the latent variable y by enumerating the two possible values ($y = 0$ and $y = 1$) of the posterior parameter $\boldsymbol{\pi}_{y|\boldsymbol{x}}$, which also implies that we need to backpropagate through the two components, one at a time, in $\boldsymbol{\sigma}^2_{\boldsymbol{\theta}_y}$ and $\boldsymbol{\mu}_{\boldsymbol{\theta}_y}$, see Figure 2.

We train Model 1 alternating the objective function

$$\mathcal{L} = \sum_i^n \mathcal{L}_{accept}((\boldsymbol{x},y)_i;\boldsymbol{\theta},\boldsymbol{\phi}) - \alpha\cdot\log\mathbb{E}_{\hat{p}(\boldsymbol{x},y)}[q_{\boldsymbol{\phi}}(y_i|\boldsymbol{x}_i)] + \sum_j^{n+m}\mathcal{L}_{reject}(\boldsymbol{x}_j;\boldsymbol{\theta},\boldsymbol{\phi}), \tag{12}$$

where $\mathbb{E}_{\hat{p}(\boldsymbol{x},y)}$ is the empirical distribution.

Note that we introduce the term $\log\mathbb{E}_{\hat{p}(\boldsymbol{x},y)}[q_{\boldsymbol{\phi}}(y_i|\boldsymbol{x}_i)]$, which is actually the classifier in Model 1, into the supervised lower bound to take advantage of the accepted applications and train the best possible classifier. The term $\alpha = \beta\cdot\frac{m+n}{n}$ controls the importance of the classification in the supervised loss function, where m and n are the number of rejected and accepted observations respectively, and β is just a scaling factor.

3.2.1 Reject Inference in Credit Scoring with Model 1

Model 1 does not just learn the distribution $p(\boldsymbol{x}|\boldsymbol{z})$ of the customers' data used in credit scoring, but it also learns a latent representation $p(\boldsymbol{z}|\boldsymbol{x}, y)$ of it. This latent representation reflects an intrinsic structure or the semantics of the customers' data. Additionally, Model 1 approximates the posterior class label distribution $q(y|\boldsymbol{x})$, which we use to estimate the default probability for new applications. This probability is given by the mutually exclusive outcomes in the posterior parameter $\boldsymbol{\pi}_{y|\boldsymbol{x}}$, which is parametrized by an MLP with softmax activation function in the output layer.

The most important characteristic of Model 1 for reject inference in credit scoring is that the unknown creditworthiness is evaluated by considering the two possible states $y = 1$ and $y = 0$ that the loan might have taken in case that the credit had been granted (Equation 10). This means that this method clearly differs from all extrapolation approaches for reject inference. Further, it is not as restrictive as the expectation-maximization algorithm since it relies on the approximation of the posterior distributions.

It can be shown that Equation 12 includes the term $KL[q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}, y)||p_{\boldsymbol{\theta}}(\boldsymbol{z}|y)]$. Then, the optimization of the objective function forces $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}, y)$ to be as close as possible to $p_{\boldsymbol{\theta}}(\boldsymbol{z}|y)$, which we have modeled as a mixture of Gaussian distributions. The first motivation for this is that the data for the accepted and rejected applications are generated by two different process, just as in [24]. Second, this mixture model generates a flexible latent space, which helps to improve the approximation of the inference process in Model 1.

Finally, the objective function in Equation 12 includes the MLP weights $\boldsymbol{\theta}$ for the densities $p(\boldsymbol{z}|y)$ and $p(\boldsymbol{x}|\boldsymbol{z})$, and $\boldsymbol{\phi}$ for the densities $q(y|\boldsymbol{x})$ and $q(\boldsymbol{z}|\boldsymbol{x}, y)$. These are all the weights in Model 1 and are present in both the supervised and unsupervised loss. Hence, the stochastic gradient optimization updates these weights jointly and estimates the different parameters $\boldsymbol{\mu}$, $\boldsymbol{\sigma}^2$, and $\boldsymbol{\pi}$ in Equation 6 and 7. In practice, when a labeled (accepted) observation is presented to the algorithm, the loss function in the backpropagation algorithm is $\mathcal{L}_{accept}((\boldsymbol{x}, y)_i; \boldsymbol{\theta}, \boldsymbol{\phi})$. Similarly, when handling unlabeled (rejected) observations the loss function is $\mathcal{L}_{reject}(\boldsymbol{x}_j; \boldsymbol{\theta}, \boldsymbol{\phi})$. In any case, all the MLP weights $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ are updated at each iteration since the same MLP handles both accepted and rejected applications.

3.3 Model 2: Generative and inference processes

Inspired by the work by [8, 16], we develop an extension of Model 1 introducing auxiliary variables. Auxiliary variables improve the variational approximation and introduce a layer of latent variables to the model's classifier. Hence, our proposed Model 2 combines a Gaussian mixture with auxiliary variables in a semi-supervised framework for the first time in the literature.

Specifically, we assume the generative process $p(\boldsymbol{x}, y, \boldsymbol{z}, \boldsymbol{a}) = p(\boldsymbol{a})p(y)p(\boldsymbol{z}|y)p(\boldsymbol{x}|\boldsymbol{z}, y)$ with the following distributions

$$\begin{aligned} p(y) &\sim \text{Bernoulli}(y; \pi), \\ p(\boldsymbol{a}) &\sim \mathcal{N}(\boldsymbol{a};\ \mathbf{0}, \mathbf{1}), \\ p(\boldsymbol{z}|y) &\sim \mathcal{N}(z|y = k;\ \boldsymbol{\mu}_{z_k} = f_{\boldsymbol{\theta}}(y), \boldsymbol{\sigma}^2_{z_k}\mathbf{I} = f_{\boldsymbol{\theta}}(y))\ for\ k = 0, 1, \\ p(\boldsymbol{x}|\boldsymbol{z}, y) &\sim \mathcal{N}(\boldsymbol{x}|\boldsymbol{z}, y;\ \boldsymbol{\mu}_x = f_{\boldsymbol{\theta}}(\boldsymbol{z}, y), \boldsymbol{\sigma}^2_x\mathbf{I} = f_{\boldsymbol{\theta}}(\boldsymbol{z}, y)). \end{aligned} \tag{13}$$

Here $\mathcal{N}$ is the Gaussian distribution and $f(\cdot)$ is a multilayer perceptron model with weights denoted by $\boldsymbol{\theta}$. The inference process factorizes as $q(\boldsymbol{z}, \boldsymbol{a}, y|x) = q(\boldsymbol{a}|\boldsymbol{x})q(y|\boldsymbol{x}, \boldsymbol{a})q(\boldsymbol{z}|\boldsymbol{x}, y)$. The distributions for this process are

$$\begin{aligned} q(\boldsymbol{a}|\boldsymbol{x}) &\sim \mathcal{N}(\boldsymbol{a}|\boldsymbol{x};\ \boldsymbol{\mu}_a = f_{\boldsymbol{\phi}}(\boldsymbol{x}), \boldsymbol{\sigma}^2_a\mathbf{I} = f_{\boldsymbol{\phi}}(\boldsymbol{x})), \\ q(y|\boldsymbol{x}, \boldsymbol{a}) &\sim \text{Bernoulli}(y|\boldsymbol{x}, \boldsymbol{a};\ \boldsymbol{\pi}_{y|\boldsymbol{x},\boldsymbol{a}} = f_{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{a})), \\ q(\boldsymbol{z}|\boldsymbol{x}, y) &\sim \mathcal{N}(\boldsymbol{z}|\boldsymbol{x}, y;\ \boldsymbol{\mu}_z = f_{\boldsymbol{\phi}}(\boldsymbol{x}, y), \boldsymbol{\sigma}^2_z\mathbf{I} = f_{\boldsymbol{\phi}}(\boldsymbol{x}, y)). \end{aligned} \tag{14}$$

Again $\mathcal{N}$ is the Gaussian distribution and $f(\cdot)$ is a multilayer perceptron model with weights denoted by $\boldsymbol{\phi}$.

Labeled data: Deriving the objective function $\mathcal{L}_{accept}$

Following the steps in Section 3.1, it is straightforward to show that the supervised negative lower bound is

$$\begin{aligned}-\mathcal{L}(\boldsymbol{x}, y; \boldsymbol{\theta}, \boldsymbol{\phi})_{accept} &= \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z},\boldsymbol{a}|\boldsymbol{x},y)}\Big[\log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}, y, \boldsymbol{z}, \boldsymbol{a})}{q_{\boldsymbol{\phi}}(\boldsymbol{z}, \boldsymbol{a}|\boldsymbol{x}, y)}\Big] \\ &= \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z},\boldsymbol{a}|\boldsymbol{x},y)}[\log p(\boldsymbol{a}) + \log p(y) + \log p_{\boldsymbol{\theta}}(\boldsymbol{z}|y) + \log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}, y) \\ &\quad - \log q_{\boldsymbol{\phi}}(\boldsymbol{a}|\boldsymbol{x}) - \log q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}, y)]. \end{aligned} \tag{15}$$

Using Equations 13 and 14 and taking the corresponding expectations, see Section B.4 in the Appendix, we obtain the lower bound for the i'th data point, as follows[8]

$$\begin{aligned}-\mathcal{L}_{accept}((\boldsymbol{x}, y)_i; \boldsymbol{\theta}, \boldsymbol{\phi}) =& \frac{1}{2}\Big[\sum_{j=1}^{\ell_z}(1 + \log \sigma^2_{\boldsymbol{\phi}_{\boldsymbol{z}_j}}) - \sum_{j=1}^{\ell_z}\Big(\log \sigma^2_{\boldsymbol{\theta}_{j,y}} + \frac{\sigma^2_{\boldsymbol{\phi}_{\boldsymbol{z}_j}}}{\sigma^2_{\boldsymbol{\theta}_{j,y}}} + \frac{(\mu_{\boldsymbol{\phi}_{\boldsymbol{z}_j}} - \mu_{\boldsymbol{\theta}_{j,y}})^2}{\sigma^2_{\boldsymbol{\theta}_{j,y}}}\Big)\Big] + \log \pi_i \\ &+ \frac{1}{2}\sum_{c=1}^{\ell_a}(\sigma^2_{\boldsymbol{\phi}_{\boldsymbol{a}_c}} + \mu^2_{\boldsymbol{\phi}_{\boldsymbol{a}_c}} - (1 + \log \sigma^2_{\boldsymbol{\phi}_{\boldsymbol{a}_c}})) + \frac{1}{L_z}\sum_{l=1}^{L_z}\log \mathcal{N}(x_i|z_{i,l}, y). \end{aligned} \tag{16}$$

Here ℓ_z and ℓ_a are the dimensions of $\boldsymbol{z}$ and $\boldsymbol{a}$ respectively, $\sigma^2_{\cdot_j}$ and $\mu_{\cdot_j}$ are the j'th element of $\boldsymbol{\sigma}^2_{\cdot}$ and $\boldsymbol{\mu}_{\cdot}$ respectively, and they refer to the variance or expectation of either $\boldsymbol{z}$ or $\boldsymbol{a}$, π_i is the prior distribution over the class label y_i, and L_z is the number of $\boldsymbol{z}_{i,l}$ samples drawn from $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}, y)$. Note that y is known in this case, hence we only backpropagate through its corresponding Gaussian component, just as in Model 1. This is specified by the subscript y in Equation 16.

Unlabeled data: Deriving the objective function $\mathcal{L}_{reject}$

Using the factorization in Equation 13 and 14, the unsupervised negative lower bound in Model 2 has the form

$$\begin{aligned}-\mathcal{L}_{reject}(\boldsymbol{x}; \boldsymbol{\theta}, \boldsymbol{\phi}) &= \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z},\boldsymbol{a},y|\boldsymbol{x})}\Big[\log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}, y, \boldsymbol{z}, \boldsymbol{a})}{q_{\boldsymbol{\phi}}(\boldsymbol{z}, \boldsymbol{a}, y|\boldsymbol{x})}\Big] \\ &= \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z},\boldsymbol{a},y|\boldsymbol{x})}[\log p(\boldsymbol{a}) + \log p(y) + \log p_{\boldsymbol{\theta}}(\boldsymbol{z}|y) + \log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}, y) \\ &\quad - \log q_{\boldsymbol{\phi}}(\boldsymbol{a}|\boldsymbol{x}) - \log q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}, y) - \log q_{\boldsymbol{\phi}}(y|\boldsymbol{x}, \boldsymbol{a})]. \end{aligned} \tag{17}$$

For the i'th observation, Equation 17 takes the following form, see Section B.5 in the Appendix,

$$\begin{aligned}-\mathcal{L}_{reject}(\boldsymbol{x}_i; \boldsymbol{\theta}, \boldsymbol{\phi}) =& \frac{1}{2}\frac{1}{L_a}\frac{1}{L_z}\sum_{l_a=1}^{L_a}\sum_{y=0}^{1}\pi_{y|\boldsymbol{x}_i,\boldsymbol{a}_{i,l_a}}\Big[\sum_{j=1}^{\ell_z}(1 + \log \sigma^2_{\boldsymbol{\phi}_{\boldsymbol{z}_j}}) - \sum_{j=1}^{\ell_z}\Big(\log \sigma^2_{\boldsymbol{\theta}_{j,y}} + \frac{\sigma^2_{\boldsymbol{\phi}_{\boldsymbol{z}_j}}}{\sigma^2_{\boldsymbol{\theta}_{j,y}}} \\ &+ \frac{(\mu_{\boldsymbol{\phi}_{\boldsymbol{z}_j}} - \mu_{\boldsymbol{\theta}_{j,y}})^2}{\sigma^2_{\boldsymbol{\theta}_{j,y}}}\Big) + \frac{1}{L_z}\sum_{l_z=1}^{L_z}\log \mathcal{N}(\boldsymbol{x}_i|\boldsymbol{z}_{i,l_z}, y_{l_a})\Big] + \frac{1}{2}\sum_{c=1}^{\ell_a}\Big(\sigma^2_{\boldsymbol{\phi}_{\boldsymbol{a}_c}} + \mu^2_{\boldsymbol{\phi}_{\boldsymbol{a}_c}} \\ &-(1 + \log \sigma^2_{\boldsymbol{\phi}_{\boldsymbol{a}_c}})\Big) + \frac{1}{L_a}\sum_{l_a=1}^{L_a}\sum_{y=0}^{1}\pi_{y|\boldsymbol{x}_i,\boldsymbol{a}_{i,l_a}}(-\log q(y|\boldsymbol{x}_i, \boldsymbol{a}_{i,l_a})) + \log \pi_i. \end{aligned} \tag{18}$$

Here all parameters are just as in $-\mathcal{L}_{accept}(\boldsymbol{x}, y; \boldsymbol{\theta}, \boldsymbol{\phi})$. It is important to note that the posterior probability over the class labels $\boldsymbol{\pi}_{y|\boldsymbol{x},\boldsymbol{a}} = [\pi_{y=0|\boldsymbol{x},\boldsymbol{a}}\ (1 - \pi_{y=0|\boldsymbol{x},\boldsymbol{a}})]$ depends on the sampled auxiliary variables. We denote this dependency explicitly using the subscript $\boldsymbol{a}$.

Finally, just as we did in Model 1, we include the term $\log q_{\boldsymbol{\phi}}(y|\boldsymbol{x}, \boldsymbol{a})$ in the unsupervised objective function to take advantage of the accepted applications. Therefore, the final objective function for Model 2 is

$$\mathcal{L} = \sum_{i}^{m}\mathcal{L}_{accept}((\boldsymbol{x}, y)_i; \boldsymbol{\theta}, \boldsymbol{\phi}) - \alpha \cdot \log \mathbb{E}_{\hat{p}(\boldsymbol{x},y,\boldsymbol{a})}[q_{\boldsymbol{\phi}}(y_i|\boldsymbol{x}_i, \boldsymbol{a}_i)] + \sum_{j}^{n}\mathcal{L}_{reject}(\boldsymbol{x}_j; \boldsymbol{\theta}, \boldsymbol{\phi}). \tag{19}$$

[8] We clutter the notation by adding the subscript $\boldsymbol{a}$ and $\boldsymbol{z}$ in the distribution parameters. This helps to differentiate the parameters of the density $q_{\boldsymbol{\phi}}(\boldsymbol{a}|\boldsymbol{x})$ from the ones in $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}, y)$.

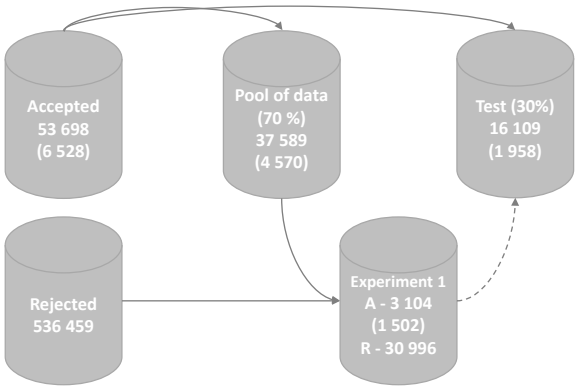


Figure 3: Data partition used in the experiments in Table 3 for the Lending Club data set. Numbers in parentheses are the number of defaulted observations, and numbers in parenthesis in percentage are the proportion of accepted applications. The experiments with the Santander data set and in Table 4 follows the same logic, but in the last sampling ('Experiment 1' box) we sample the number of accepted and rejected applications as needed.

3.3.1 Reject Inference in Credit Scoring with Model 2

Model 2 has almost the same characteristics as Model 1, but there are two new items. First, Model 2 approximates two layers of latent representations $q(\boldsymbol{a}|\boldsymbol{x})$ and $q(\boldsymbol{z}|\boldsymbol{x}, y)$. The posterior distribution $q(\boldsymbol{a}|\boldsymbol{x})$, together with the customers' data $\boldsymbol{x}$, is used to estimate the default probability (Equation 14). By doing so, Model 2 has a relatively more expressive estimation of creditworthiness. The presumption is that the latent representation $\boldsymbol{a}$ captures the intrinsic structure of the data and that it therefore provides relevant features for enhancing the performance of the classifier $q(y|\boldsymbol{x}, \boldsymbol{a})$. Finally, note that $q(\boldsymbol{a}|\boldsymbol{x})$ is assumed to be multivariate Gaussian distributed, hence we use the reparametrization trick (see Section 3.2) to sample from this distribution, i.e. $\boldsymbol{a} = \boldsymbol{\mu}_a + \boldsymbol{\sigma}_a \odot \boldsymbol{\epsilon}$ where $\boldsymbol{\mu}_a$ and $\boldsymbol{\sigma}_a$ are the outputs in the MLP for the density $q(\boldsymbol{a}|\boldsymbol{x})$.

The second difference from Model 1 is that the data generating process $p(\boldsymbol{x}|\boldsymbol{z}, y)$ is conditioned on the latent variable $\boldsymbol{z}$ and class label y. This is simply done to achieve better training stability. See Section 4.3 for more details about model training.

4 Experiments and Results

The goal with the experiments is twofold. First, we compare the performance of our proposed models with a range of techniques representing the state-of-the-art in reject inference for credit scoring, including three classical reject inference techniques (reclassification, fuzzy parceling and augmentation [17]) and three semi-supervised machine learning approaches (self-learning [18] MLP, self-learning SVM, and semi-supervised SVM [19]) under a realistic scenario preserving the original acceptance rates in two real data sets. Second, to have a better understanding of the behaviour of reject inference models for credit scoring, we test the model performance in different scenarios varying the number of accepted and rejected observations. In both cases, we include two supervised machine learning models (multilayer perceptron (MLP) [20] and support vector machine (SVM) [21]) to measure the marginal gain of reject inference.

4.1 Data description

We use two real data sets containing both rejected and accepted applications. The first data set is public[9] and consists of personal loan applications through Lending Club, which is the world's largest peer-to-peer lending company. We replicate the data sample used in [23], which includes applications

[9]The data can be obtain directly at the Lending Club's website, however they require the user to login. We obtain a complete version of the available data at the website https://github.com/nateGeorge/preprocess_lending_club_data, which is updated quarterly.

Lending Club and Santander Credit Cards	
MLP Network	Number of hidden layers and dimensions
$q(\boldsymbol{z}\|\boldsymbol{x}, y)$	[10 10]*, [10 20], [10 30], [10 50], [100 70]***, [10 20 10], [10 30 10], [10 40 10]**, [10 50 10], [60 90 60]****
$p(\boldsymbol{x}\|\cdot)$	[10 10]*, [10 20], [10 30], [10 50], [70 100]***, [10 20 10], [10 30 10], [10 40 10]**, [10 50 10], [60 90 60]****
$p(\boldsymbol{z}\|y)$	[10]*,**,***,****
$q(\boldsymbol{a}\|\boldsymbol{x})$	[50], [10 10], [10 20], [10 30], [10 40]**, [10 50], [20 40], [20 50], [30 50], [30 60], [40 60]****
$q(y\|\cdot)$	[50], [60], [70]*,[80]***, [100]****, [120], [130]**
Parameter/hyperparameter	Value
$\boldsymbol{z}$ dimension	30, 50*,**,****, 100***
$\boldsymbol{a}$ dimension	30, 50**,****
β	0.008**, 0.01, 0.025, 0.14, 1.1*, 3****, 8***

Table 2: Grid for hyperparameter optimization for Model 1 and 2 and for both data sets. The numbers within brackets specify the number of neurons in each hidden layers, i.e. [10 10] means two hidden layers with 10 neurons each. Finally, the superscript * and ** shows the final architecture for Model 1 and Model 2 respectively for the Lending Club data set used in Table 3. Similarly, *** and **** shows the final architecture for Model 1 and Model 2 respectively for the Santander Credit Cards data set used in Table 3.

from January 2009 until September 2012 with 36-months maturity. However, we do not split the data set in yearly sub samples, since we want to keep as many observations from the minority class ($y = 1$) as possible. Hence, the data set that we use in our experiments has 53 698 accepted applications, including 6 528 defaults, and 536 459 rejected applications[10]. That is, the acceptance ratio is 9.10% and default rate is 12.16%. For more details about the Lending Club data, see Table A1 in the Appendix.

The second data set is provided by Santander Consumer Bank Nordics and consists of credit card applications arriving trough their internet website. The applications were received during the period January 2011 until December 2016. During this period Santander accepted 126 520 applications and only 14 993 customers ended up as defaults. The number of rejected applications during this period is 232 898. Hence, the acceptance ratio is 35.20% and default rate 11.85%.

In addition to these two data sets, we have two small samples after September 2012 and December 2016 for Lending Club and Santander Bank respectively, which are used to produce well-calibrated estimates of class probabilities using the beta calibration approach [49]. These samples are not part of the experimental design explained in Section 4.2.

4.2 Experimental Design

We conduct two different set of experiments. In the first experimental setup, we keep the original acceptance ratio, but we do not use more than 34 100 observations in total [11]. To construct this data set, we first split the original data in 70%-30% for training and testing respectively. Then, we down sample the majority class ($y = 0$) in the training set until it equals the number of observations for the minority class ($y = 1$). To achieve the correct acceptance ratio, this requires a random selection of both class labels. Note that the test data set is left as it is, i.e. it preserves the original default rate. Finally, we randomly select the number of reject applications in a way that these, together with the balanced training sample, do not exceed 34 100 observations, see Figure 3.

In the second set of experiments[12], we analyze the effect of varying the number of accepted (rejected) applications, while keeping the same number of rejected (accepted) applications. We follow the same approach as in the the first experiments, splitting the data set into a training and test data set, down sampling the training set, and randomly selecting the number of reject applications.

For the Lending Club data set, we use all variables in Table A1 to train all models, while for the Santander data we use a forward selection approach to select the explanatory variables that are included in the

[10]The number of accepted and rejected applications are not exactly the same as in [23], but the variable statistics are very similar and the default trend is the same. See Table A1 for more information.

[11]This is done to allow a fair comparison to S3VM, which does not scale to larger datasets due to memory requirements. For the 34 100 observations, S3VM requires 123GB of memory to estimate the kernel matrix.

[12]S3VM is not included in this section since it takes around 356 hours to evaluate each scenario in this section and in total we evaluate 12 different scenarios. In addition, it has the memory restrictions already mentioned. Similarly, the iterative procedure in the self-learning SVM is not feasible in this section.

	Lending Club (LC)					Santander Credit Cards (SCC)					Runtime	
	AUC	GINI	H-measure	Recall	Precision	AUC	GINI	H-measure	Recall	Precision	LC	SCC
MLP	0.6273	0.2547	0.0535	0.4454	0.1738	0.7091	0.4183	0.1326	0.7909	0.1772	00:01.28	00:04.53
SVM	0.6284	0.2567	0.0543	0.4632	0.1783	0.7388	0.4777	0.1689	0.7997	0.1895	00:06.59	00:14.42
Reclassification	0.5784	0.1567	0.0227	0.4906	0.1493	0.6415	0.2830	0.0625	0.9989	0.1187	00:05.04	00:01.15
Fuzzy Parceling	0.6198	0.2560	0.0540	0.4598	0.1772	0.6791	0.3582	0.0957	0.8676	0.1541	00:03.82	00:08.45
Augmentation	0.6219	0.2558	0.0541	0.4581	0.1777	0.6761	0.3523	0.0923	0.8735	0.1524	00:13.07	00:15.25
Self-lerning MLP	0.5868	0.1737	0.0326	0.4504	0.1570	0.6726	0.3451	0.0877	0.8502	0.1519	00:18.80	00:20.53
Self-lerning SVM	0.6206	0.2551	0.0535	**0.4957**	0.1731	0.7266	0.4532	0.1529	0.8494	0.1725	03:25.89	05:08.36
S3VM	0.6201	0.2402	0.0481	0.0000	NA	0.6520	0.3040	0.0733	**1.0000**	0.1185	09:17.00	06:20.12
Model 1	0.6294	0.2588	0.0554	0.4540	0.1788	0.7394	0.4788	0.1678	0.8326	0.1848	10:48.19	04:12.16
Model 2	**0.6363**	**0.2755**	**0.0632**	0.4688	**0.1825**	**0.7431**	**0.4851**	**0.1764**	0.6282	**0.2303**	12:24.06	05:54.33

Table 3: Model performance keeping the original acceptance ratios, i.e. 9.10% for Lending Club (LC) and 35.20% for Santander Credit Cards (SCC). The training data set is balanced by down sampling the majority class, and the threshold used to calculate recall and precision is based on the empirical default rate in the test data set. The last two columns show the runtime for one cross-validation and the format is given in mm:ss.cs, where mm, ss, and cs stands for minutes, seconds and centiseconds respectively.

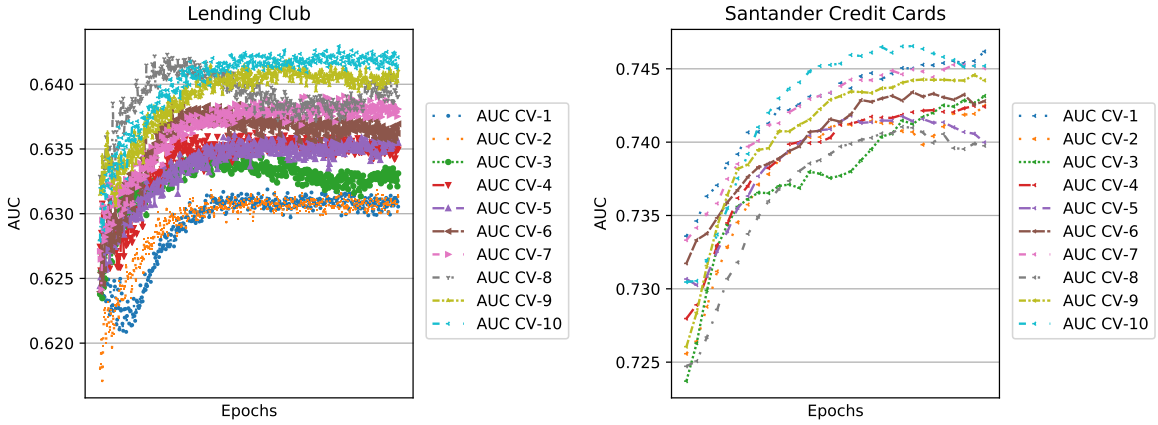


Figure 4: The left panel shows the AUC performance for the Lending Club data set in the 10 cross-validations (CV), while the right panel shows the performance for the Santander Bank data set. Both diagrams correspond to Model 2.

reclassification, fuzzy parceling and augmentation methods[13]. For the other models we use all variables in Table A2. Finally, we do hyperparameter tuning using grid search with 10-cross validation for the MLP, SVM, S3VM, Model 1, and Model 2. The best architecture for the MLP and SVM is used as the base model in the self-training approaches for MLP and SVM. The details of the grid search are given in Table A3.

4.3 Model Implementation and Training

Model 1 and Model 2 are implemented in Theano [50]. We use softplus activation functions in all hidden layers and linear activation functions in all output layers estimating $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}^2$. For the output layer in the classifiers $q_{\boldsymbol{\phi}}(y|\cdot)$ we use softmax activation functions. Further, we use the Adam optimizer [51] with learning rate equal to 1e-4 and 5e-5 for training of Model 1 and Model 2 respectively. The rest of parameters in the Adam optimizer are the default values suggested in the original paper. We use $L = 1$ and $L_a = 1$ for both Model 1 and 2 in all experiments. Finally, both data sets are standardized before training and testing, and the class label y is one-hot-encoded. The model architectures used in the experiments in Table 3 are shown in Table 2.

It is important to mention that deep generative models are, in general, difficult to train [52, 53]. The training of Model 1 and Model 2 in some cases become unstable, especially for the experiments where we vary the number of accepted and rejected applications. Moreover, it is sensitive to the initial weights. Hence, we use a Variational Autoencoder [45] to pretrain the weights in $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}, y)$ and $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$ for Model

[13]These three methods are based on the logistic regression. Hence, the forward selection approach prevents the logistic regression from overfitting and avoids numerical problems on its optimization.

Lending Club												
	Accepted applications						Rejected applications					
No. observations	200 (0.04%)	600 (0.11%)	1 200 (0.22%)	2 000 (0.37%)	6 000 (1.11%)	All (1.67%)	30 997 (0.64%)	100 000 (0.20%)	200 000 (0.10%)	300 000 (0.07%)	400 000 (0.05%)	All (0.04%)
MLP	0.6002	0.6236	0.6237	0.6304	0.6299	0.6307	0.6037	0.6037	0.6037	0.6037	0.6037	0.6037
SVM	0.6039	0.6267	0.6253	0.6320	0.6302	0.6309	0.6054	0.6054	0.6054	0.6054	0.6054	0.6054
Reclassification	0.5786	0.5785	0.5812	0.5853	0.5806	0.5816	0.5616	0.5785	0.5783	0.5574	0.5693	0.5779
Fuzzy Parceling	0.6017	0.6240	0.6232	0.6295	0.6297	0.6302	0.6041	0.6026	0.6018	0.6031	0.6073	0.6006
Augmentation	0.6017	0.6216	0.6207	0.6301	0.6295	0.6304	0.6023	0.6028	0.6010	0.5967	0.5953	0.5979
Self-lerning MLP	0.5824	0.5728	0.5734	0.5675	0.5858	0.5631	0.5640	0.5485	0.5706	0.5715	0.5758	0.5703
Model 2	**0.6175**	**0.6269**	**0.6310**	**0.6344**	**0.6381**	**0.6404**	**0.6112**	**0.6075**	**0.6091**	**0.6107**	**0.6121**	**0.6175**
Runtime												
Self-learning MLP	00:20:36	00:26:14	00:29:31	00:29:23	00:31.39	00:35:11	00:02.10	00:05:02	00:09:50	00:15:01	00:18:02	00:23:36
Model 2	02:39:02	02:41:75	02:55:19	03:24:13	03:42:17	04:03:10	00:14:18	00:38:07	01:09:02	01:39:48	02:00:54	02:39:02

Table 4: Left panel: Model performance, measured with AUC, as a function of accepted applications. In all six experiments to the left, we use all 536 459 rejected applications. Right panel: Model performance, measured with AUC, as a function of rejected applications. In all six experiments to the right, we use only 200 accepted applications. Numbers in parenthesis are the acceptance ration for each experiment. The last two rows show the runtime for one cross-validation and the format is in hh:mm:ss, where hh, mm, and ss stands for hours, minutes, and seconds respectively. We do not include the runtime for the first five models because the difference with respect to the runtimes in Table 3 is negligible.

1. Similarly, we prewarm all weights $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ in Model 2. In both cases, we initialized the MLP weights as suggested in [54]. We also achieve more stable training in Model 2 by conditioning $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}, y)$ on the class label y.

4.4 Benchmark Reject Inference

Table 3 compares the performance of Model 1 and Model 2 with other models when using the original acceptance ratio in the data sets. It can be seen that both Model 1 and Model 2 perform better than all supervised and semi-supervised models in terms of AUC, GINI, H measure and precision. Our results support previous findings that the reclassification, fuzzy parcelling and augmentation methods do not improve model performance. The reclassification approach is consistently the worst model. Further, the self-training approaches do not improve the performance of the base models MLP and SVM. Finally, S3VM has significantly worse performance than the base models for the Santander Credit Cards data set.

We use the Platt scaling method [55] to get (pseudo) default probabilities from SVM and S3VM. It is interesting to see that we could not estimate the recall and precision for S3VM in the Lending Club data because the estimated default probabilities are concentrated around the average, with practically no dispersion, see Table A4. S3VM estimates default probabilities for all applications below the default rate in the Lending Club data set, and above the default rate in the Santander data set.

Model 2 performs better than Model 1 in terms of all measures except for recall. Remember that the main difference between these models is the classifier in Model 2, which uses a latent representation of the customers' data. Our results are hence in correspondence with previous studies showing the predictive power embedded in the latent transformations. It is further interesting to note that our proposed models for reject inference not only perform better, but also estimate higher variability in the predicted default probabilities, as shown in Table A4. This result supports previous findings that the default probability is underestimated if reject inference is ignored. Unfortunately, given the nature of the data sets in this research we are not able to draw any conclusion about the economic impact of this interesting detail.

It is worth mentioning that Model 2 is the algorithm that takes longer time to converge for the Lending Club data set, while for the Santander Credit Cards data set is S3VM. In any case, the runtime for both Model 2 and S3VM, in the experiments in Table 3, is moderate.

In Table 4, we analyze the impact of the number of accepted and rejected applications on model performance using Model 2 and the Lending Club data set. In the right panel, we can observe that the general trend is that the more rejected applications we add to Model 2, the higher model performance. In the left panel, we can see that the more accepted data we have available, the better model performance for the supervised models and the less difference compared to Model 2. Note that Model 2 achieves the

highest average AUC of 0.6404 in the *All* scenario, which includes 545 599 observations. This is 16 times more data compared to what self-training SVM and S3VM handled.

The runtime for Model 2 in the experiments that use all rejected applications has increased significantly compared to Table 3. In the scenario where we use all accepted and rejected applications, 545 599 observations in total, Model 2 takes about 4 hours to converge. Note that this model has 16 080 learnable parameters, which are significantly more than the 502 parameters in the MLP. Generally, training deep learning architectures is computationally intensive and the computational complexity increases linearly with the number of parameters (including MLP architectures). However, training can be accelerated by distributing training in parallel across multiple GPUs.

5 Conclusion

In this research we develop two novel deep generative models for reject inference in credit scoring. Our models use the posterior distribution of the outcome of the loan to infer the unknown creditworthiness of the rejected applications. This is done by exact enumeration of the two possible outcomes of the loan, which is an advantage compared to reject inference methods based on extrapolation. To the best of our knowledge, this is the first research that develops novel methods for reject inference in credit scoring coupling Gaussian mixtures and auxiliary variables in a semi-supervised framework with generative models.

The experiments show that our proposed models achieve higher model performance compared to many of the classical and machine learning approaches for reject inference in credit scoring, and the models' performance increases as we add more data for model training. Further, the efficient stochastic gradient optimization technique used in deep generative models scales to large data sets, which is an advantage over supervised and semi-supervised support vector machines. Note that even though the focus of this research is on credit scoring, our proposed models generalize to other research domains, e.g. image classification.

The higher model performance of our proposed methodology is further enhanced by adding latent representations of the customers' data to the classifier. This data representation captures the intrinsic structure of the data providing relevant information for classification. Since our proposed approach for reject inference in credit scoring offers flexible modeling possibilities, we hope that this research spurs future work on reject inference in credit scoring using deep generative model focusing on improving the training stability and classification power.

Acknowledgements

The authors would like to thank Santander Consumer Bank for financial support and the real data set used in this research. This work was also supported by the Research Council of Norway [grant number 260205] and SkatteFUNN [grant number 276428].

6 Appendix

A Tables and Figures

To replicate the data set presented in [23], we excluded all observations with missing values in any of the variables in Table A1. Further, the allowed variable range, which we choose based on [23], is determined by the minimum and maximum values as shown in the table. The summary statistics in our data sample is not exactly the same as in [23], but the default trend is the same (the default rate in 2009 is 12.59%, 2010 is 9.61%, 2011 is 10.32% and in 2012 is 13.76%).

Table A1: Lending Club Descriptive Statistics

	Variable	Mean	Std	Min	1 Quantile	Median	3 Quantile	Max
Accepts	Debt to income	14.51	7.19	0.00	9.06	14.44	19.82	34.99
	Loan amount	10 610.34	6 738.61	1 000.00	5 706.25	9 600.00	14 000.00	35 000.00
	Fico score	711.49	35.06	662.00	682.00	707.00	732.00	847.50
	State d1	0.43	0.49	0.00	0.00	0.00	1.00	1.00
	State d2	0.43	0.49	0.00	0.00	0.00	1.00	1.00
	State d3	0.10	0.29	0.00	0.00	0.00	0.00	1.00
	Employment length	3.97	3.18	0.00	1.00	3.00	6.00	10.00
Rejects	Debt to income	24.29	31.14	0.00	7.90	18.19	31.18	419.33
	Loan amount	13 330.74	10 361.51	1 000.00	5 000.00	10 000.00	20 000.00	35 000.00
	Fico score	638.15	74.10	385.00	595.00	651.00	690.00	850.00
	State d1	0.47	0.50	0.00	0.00	0.00	1.00	1.00
	State d2	0.37	0.48	0.00	0.00	0.00	1.00	1.00
	State d3	0.10	0.30	0.00	0.00	0.00	0.00	1.00
	Employment length	8.40	3.16	0.00	10.00	10.00	10.00	10.00

The second data set which we use in this research is provided by Santander Consumer Bank. The details that we can provide about this data set are limited by its proprietary nature. The descriptive statistics are shown in Table A2.

Table A2: Santander Credit Cards Descriptive Statistics

	Variable	Mean	Std	Min	1 Quantile	Median	3 Quantile	Max
Accepts	Var1	86 475.84	107 975.22	0.00	29 852.00	69 162.00	108 898.00	10 570 323.00
	Var2	152 205.11	1 778 838.75	0.00	0.00	0.00	4 376.00	393 676 928.00
	Var3	38.95	13.38	19.00	28.00	37.00	48.00	92.00
	Var4	976 647.69	16 125 692.00	−2.00	−2.00	−2.00	1 250 000.00	2 701 061 888.00
	Var5	903 518.75	3 228 558.75	−2.00	−2.00	−2.00	1 430 000.00	985 694 976.00
	Var6	807 869.63	13 848 935.00	0.00	0.00	0.00	1 075 000.00	2 667 096 064.00
	Var7	95 622.16	14 090 133.00	−2 664 925 952.00	−2.00	−2.00	79 000.00	984 075 008.00
	Var8	9.46	23.82	−2.00	−2.00	−2.00	4.63	100.00
	Var9	−0.44	1.86	−2.00	−2.00	−2.00	1.00	82.00
	Var10	−0.91	1.14	−2.00	−2.00	−2.00	0.00	4.00
	Var11	−1.99	0.15	−2.00	−2.00	−2.00	−2.00	3.00
	Var12	−0.63	2.06	−2.00	−2.00	−2.00	1.00	164.00
	Var13	−0.34	2.09	−2.00	−2.00	−2.00	1.00	164.00
	Var14	−1.98	0.32	−2.00	−2.00	−2.00	−2.00	26.00
	Var15	−0.47	1.73	−2.00	−2.00	−2.00	1.00	52.00
	Var16	−1.15	1.00	−2.00	−2.00	−2.00	0.00	1.00
	Var17	0.16	0.53	0.00	0.00	0.00	0.00	19.00
	Var18	0.95	2.25	0.00	0.00	0.00	1.00	67.00
	Var19	1.12	2.42	0.00	0.00	0.00	1.00	72.00
	Var20	1.57	3.27	0.00	0.00	0.00	2.00	97.00
	Var21	357 123.84	372 109.81	0.00	170 103.14	295 917.44	443 333.95	34 850 852.00
	Var22	8.29	8.53	0.00	3.97	6.91	10.29	760.94
	Var23	37 156.38	250 887.75	−12 873 071.00	−14 218.19	23 241.04	79 463.82	33 829 372.00
	Var24	16 168.70	432 254.88	−40 114 780.00	0.00	0.00	0.00	50 003 248.00
	Var25	9 037.99	60 101.17	−2 641 216.00	−4 085.00	5 520.00	19 799.25	6 169 685.00
	Var26	0.35	42.04	0.00	0.20	0.23	0.26	14 940.20

Table 2 Continued

	Variable	Mean	Std	Min	1 Quantile	Median	3 Quantile	Max
	Var27	0.47	0.50	0.00	0.00	0.00	1.00	1.00
	Var28	46.04	75.70	−29.00	−2.00	12.00	65.00	754.00
	Var29	6.71	34.72	−2.00	−2.00	−2.00	−2.00	412.00
	Var30	6.71	34.72	−2.00	−2.00	−2.00	−2.00	412.00
	Var31	1.08	0.97	0.00	0.53	0.90	1.36	43.75
	Var32	0.98	1.02	0.00	0.47	0.82	1.22	101.95
	Var33	0.98	1.01	0.00	0.47	0.81	1.22	99.13
	Var34	0.56	1.18	0.00	0.00	0.00	1.00	73.00
	Var35	0.49	0.50	0.00	0.00	0.00	1.00	1.00
	Var36	0.00	0.01	0.00	0.00	0.00	0.00	1.00
	Var37	0.58	0.49	0.00	0.00	1.00	1.00	1.00
	Var38	0.07	0.25	0.00	0.00	0.00	0.00	1.00
	Var39	0.21	0.41	0.00	0.00	0.00	0.00	1.00
	Var40	0.09	0.29	0.00	0.00	0.00	0.00	1.00
	Var41	0.06	0.23	0.00	0.00	0.00	0.00	1.00
	Var42	0.01	0.11	0.00	0.00	0.00	0.00	1.00
	Var43	0.37	0.48	0.00	0.00	0.00	1.00	1.00
	Var44	0.53	0.50	0.00	0.00	1.00	1.00	1.00
	Var45	0.09	0.28	0.00	0.00	0.00	0.00	1.00
	Var46	0.00	0.02	0.00	0.00	0.00	0.00	1.00
	Var47	0.65	0.48	0.00	0.00	1.00	1.00	1.00
	Var48	0.26	0.44	0.00	0.00	0.00	1.00	1.00
	Var49	0.06	0.23	0.00	0.00	0.00	0.00	1.00
	Var50	0.00	0.00	0.00	0.00	0.00	0.00	1.00
	Var51	0.03	0.18	0.00	0.00	0.00	0.00	1.00
	Var52	0.75	0.44	0.00	0.00	1.00	1.00	1.00
	Var53	0.25	0.44	0.00	0.00	0.00	1.00	1.00
	Var54	0.08	0.27	0.00	0.00	0.00	0.00	1.00
	Var55	0.16	0.36	0.00	0.00	0.00	0.00	1.00
	Var56	0.39	0.49	0.00	0.00	0.00	1.00	1.00
	Var57	0.30	0.46	0.00	0.00	0.00	1.00	1.00
	Var58	0.08	0.27	0.00	0.00	0.00	0.00	1.00
Rejects	Var1	57 198.23	68 931.46	0.00	12 800.00	43 182.50	80 412.00	3 635 832.00
	Var2	33 128.01	568 171.50	0.00	0.00	0.00	0.00	208 626 176.00
	Var3	34.60	12.16	1.00	25.00	32.00	42.00	95.00
	Var4	507 337.69	11 648 304.00	−2.00	−2.00	−2.00	105 937.50	2 701 061 888.00
	Var5	434 133.63	1 137 152.75	−2.00	−2.00	−2.00	0.00	72 376 000.00
	Var6	432 619.66	10 198 556.00	0.00	0.00	0.00	0.00	2 303 705 088.00
	Var7	1 499.88	10 159 168.00	−2 299 855 104.00	−2.00	−2.00	−2.00	72 376 000.00
	Var8	3.45	16.70	−2.00	−2.00	−2.00	0.00	100.00
	Var9	−1.16	1.51	−2.00	−2.00	−2.00	0.00	82.00
	Var10	−1.39	1.02	−2.00	−2.00	−2.00	0.00	4.00
	Var11	−1.87	0.95	−2.00	−2.00	−2.00	−2.00	36.00
	Var12	−1.24	1.67	−2.00	−2.00	−2.00	−2.00	105.00
	Var13	−1.06	1.77	−2.00	−2.00	−2.00	1.00	105.00
	Var14	−1.79	1.20	−2.00	−2.00	−2.00	−2.00	38.00
	Var15	−1.13	1.52	−2.00	−2.00	−2.00	1.00	43.00
	Var16	−1.52	0.87	−2.00	−2.00	−2.00	−2.00	1.00
	Var17	0.26	0.74	0.00	0.00	0.00	0.00	87.00
	Var18	3.28	6.06	0.00	0.00	1.00	4.00	166.00
	Var19	3.54	6.30	0.00	0.00	1.00	4.00	172.00
	Var20	4.62	7.90	0.00	0.00	2.00	5.00	176.00
	Var21	250 519.14	242 146.78	0.00	112 918.59	212 571.75	337 357.29	13 897 584.00
	Var22	5.80	5.55	0.00	2.64	4.94	7.84	308.84
	Var23	23 313.24	179 360.19	−31 086 966.00	−15 761.49	16 862.45	61 574.02	11 590 733.00
	Var24	2 551.04	171 498.02	−30 644 804.00	0.00	0.00	0.00	16 552 538.00
	Var25	5 758.38	43 678.19	−6 499 649.00	−3 843.00	3 537.00	14 794.00	1 851 795.00
	Var26	0.30	31.14	0.00	0.16	0.23	0.26	14 940.20
	Var27	0.25	0.43	0.00	0.00	0.00	1.00	1.00
	Var28	32.24	65.42	−43.00	−2.00	−2.00	43.00	804.00
	Var29	6.67	32.32	−2.00	−2.00	−2.00	−2.00	377.00
	Var30	6.67	32.32	−2.00	−2.00	−2.00	−2.00	377.00
	Var31	0.77	0.70	0.00	0.35	0.67	1.05	36.99
	Var32	0.69	0.67	0.00	0.31	0.59	0.93	38.16

Table 2 Continued

Variable	Mean	Std	Min	1 Quantile	Median	3 Quantile	Max
Var33	0.69	0.66	0.00	0.31	0.59	0.93	38.90
Var34	0.36	1.07	0.00	0.00	0.00	0.00	97.00
Var35	0.27	0.45	0.00	0.00	0.00	1.00	1.00
Var36	0.00	0.01	0.00	0.00	0.00	0.00	1.00
Var37	0.51	0.50	0.00	0.00	1.00	1.00	1.00
Var38	0.07	0.26	0.00	0.00	0.00	0.00	1.00
Var39	0.23	0.42	0.00	0.00	0.00	0.00	1.00
Var40	0.14	0.34	0.00	0.00	0.00	0.00	1.00
Var41	0.05	0.22	0.00	0.00	0.00	0.00	1.00
Var42	0.00	0.07	0.00	0.00	0.00	0.00	1.00
Var43	0.20	0.40	0.00	0.00	0.00	0.00	1.00
Var44	0.75	0.43	0.00	0.00	1.00	1.00	1.00
Var45	0.05	0.22	0.00	0.00	0.00	0.00	1.00
Var46	0.00	0.02	0.00	0.00	0.00	0.00	1.00
Var47	0.74	0.44	0.00	0.00	1.00	1.00	1.00
Var48	0.16	0.37	0.00	0.00	0.00	0.00	1.00
Var49	0.06	0.23	0.00	0.00	0.00	0.00	1.00
Var50	0.06	0.00	0.00	0.00	0.00	0.00	1.00
Var51	0.04	0.19	0.00	0.00	0.00	0.00	1.00
Var52	0.55	0.50	0.00	0.00	1.00	1.00	1.00
Var53	0.45	0.50	0.00	0.00	0.00	1.00	1.00
Var54	0.09	0.29	0.00	0.00	0.00	0.00	1.00
Var55	0.16	0.37	0.00	0.00	0.00	0.00	1.00
Var56	0.38	0.48	0.00	0.00	0.00	1.00	1.00
Var57	0.28	0.45	0.00	0.00	0.00	1.00	1.00
Var58	0.09	0.28	0.00	0.00	0.00	0.00	1.00

Table A3: Grid for hyperparameter optimization for Lending Club: The total number of model configurations are 132, 160 and 240 for MLP, SVM, and S3VM respectively. For the Santander data set the number of model configurations evaluated are 204, 160, and 240 for MLP, SVM, and S3VM respectively.

Lending Club

MLP		SVM		S3VM	
Layers	1	C	5, 10, 13, 14, 15, 17, 19, 21, 23, 25	C	1, 5, 10, 13, 15, 17
Neurons	3, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60	Gamma	2, 1.5, 1, 0.5, 0.1, 0.01, 0.001, auto	Gamma	2.5, 2, 1.5, 1, 0.5
Activation	logistic, tanh, relu	Kernel	rbf, linear	Kernel	rbf, linear
Learning rate	constant, adaptive			LamU	0.5, 1, 1.5, 2
Solver	sgd, adam				

Santander Credit Cards

Layers	1	C	5, 10, 13, 14, 15, 17, 19, 21, 23, 25	C	1, 5, 10, 13, 15, 17
Neurons	50, 60, 65, 70, 75, 80, 85, 90, 95, 100, 105, 110, 115, 120, 130, 140, 150	Gamma	2, 1.5, 1, 0.5, 0.1, 0.01, 0.001, auto	Gamma	2.5, 2, 1.5, 1, 0.5
Activation	logistic, tanh, relu	Kernel	rbf, linear	Kernel	rbf, linear
Learning rate	constant, adaptive			LamU	0.5, 1, 1.5, 2
Solver	sgd, adam				

Table A4: Empirical moment statistic for the default probability.

	Lending Club				Santander Credit Cards			
	Average	Std.	Kurtosis	Skewness	Average	Std.	Kurtosis	Skewness
MLP	0.1101	0.0096	−0.1027	0.0969	0.1180	0.0146	−0.0885	0.0563
SVM	0.1012	0.0130	−0.1505	0.0420	0.1202	0.0199	−0.1016	0.0517
Reclassification	0.1066	0.0083	−0.0635	−0.2861	0.1200	0.0011	6.1730	−0.8207
Fuzzy Parceling	0.1003	0.0132	−0.1389	0.0813	0.1198	0.0041	0.6406	−0.6061
Augmentation	0.0995	0.0131	−0.1487	0.0881	0.1198	0.0040	0.6285	−0.6151
Self-learning MLP	0.1055	0.0116	−0.0471	0.0770	0.1276	0.0058	0.2282	−0.5179
Self-learning SVM	0.1014	0.0130	−0.1494	0.0384	0.1257	0.0147	−0.1199	−0.0741
S3VM	0.1203	1.39e-6	−0.1173	−0.1297	0.1200	7.08e-7	0.7407	0.8687
Model 1	0.0985	0.0408	−0.5650	0.3368	0.1190	0.0367	−1.1459	−0.2455
Model 2	0.0999	0.0424	−0.5366	0.3819	0.0925	0.0340	0.8182	0.7802

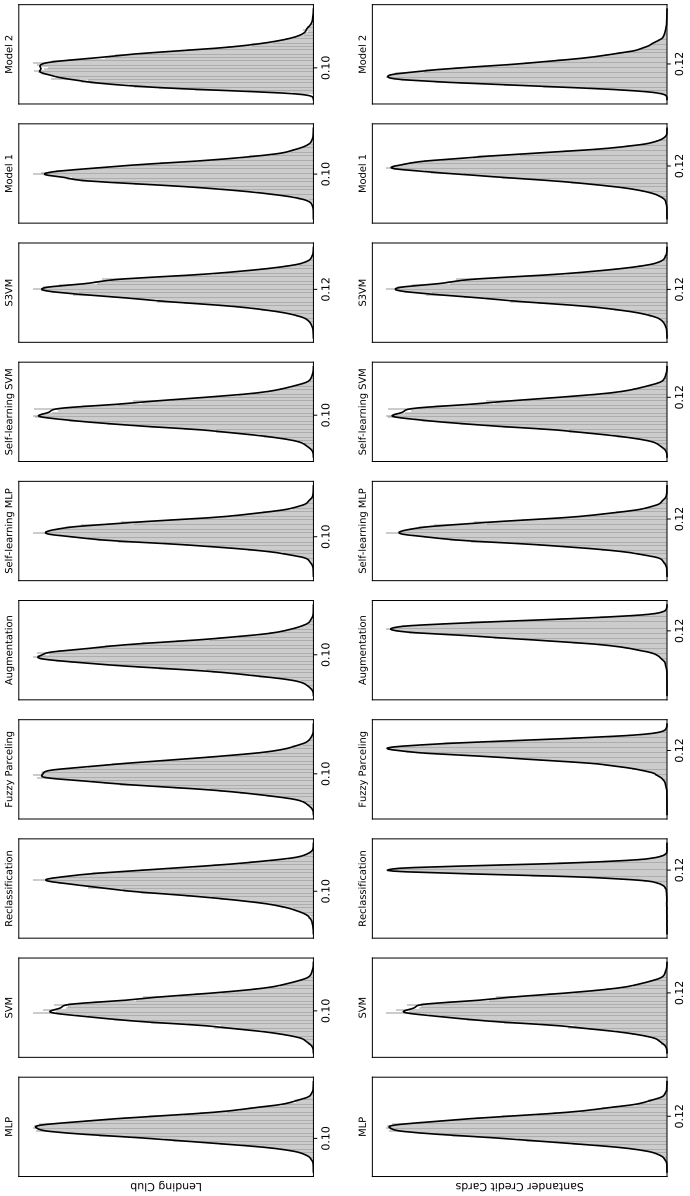


Figure A1: Empirical distribution of the default probability for the original acceptance ratio as explained in Section 4.2.

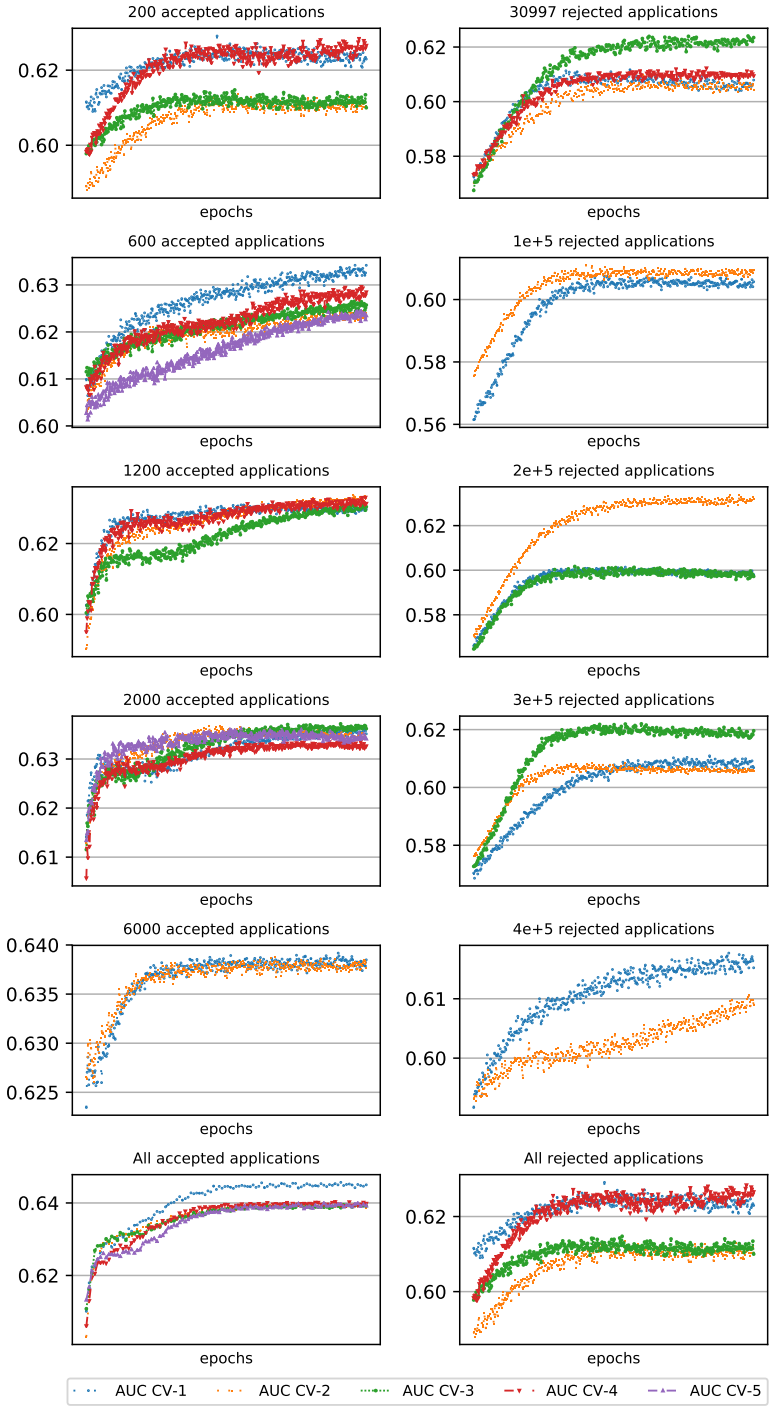


Figure A2: Model performance based on 5 cross-validations (CV) for the different scenarios analyzed in Table 4, using the Lending Club data set and Model 2. Since training for these scenarios in some cases become unstable, we keep only the results where Model 2 converged. Note that Model 2 achieves the highest AUC equal to 0.6450 in the *All* scenario in the left panel.

B Deriving the lower bounds

B.1 Lemma 1

Given two multivariate Gaussian distribution, with diagonal covariance matrix, $p(\boldsymbol{x}) \sim \mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\sigma}_1^2 \boldsymbol{I})$ and $q(\boldsymbol{x}) \sim \mathcal{N}(\boldsymbol{\mu}_2, \boldsymbol{\sigma}_2^2 \boldsymbol{I})$, where $\boldsymbol{\mu}_\cdot \in \mathbf{R}^d$ and $\boldsymbol{\sigma}_\cdot^2 \in \mathbf{R}^d$, we have:

$$\int q(\boldsymbol{x}) \ \log p(\boldsymbol{x}) d\boldsymbol{x} = \sum_{i=1}^{d} -\frac{1}{2}\log(2\pi\sigma_{1,i}^2) - \frac{\sigma_{2,i}^2}{2\sigma_{1,i}^2} - \frac{(\mu_{2,i}-\mu_{1,i})^2}{2\sigma_{1,i}^2}, \tag{B1}$$

where $\mu_{\cdot,i}$ and $\sigma_{\cdot,i}$ are the i'th element of $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}^2$ respectively.

Proof:

$$\begin{aligned}
&\int q(\boldsymbol{x}) \ \log p(\boldsymbol{x}) d\boldsymbol{x} = \int q(\boldsymbol{x}) \log \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\Big(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu_1})^T\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu_1})\Big) d\boldsymbol{x} \\
&= -\frac{1}{2}\log(2\pi\sigma_{1,i}^2) - \int q(\boldsymbol{x}) \frac{(x_i-\mu_{1,i})^2}{2\sigma_{1,i}^2} d\boldsymbol{x} - \cdots - \frac{1}{2}\log(2\pi\sigma_{1,d}^2) - \int q(\boldsymbol{x}) \frac{(x_d-\mu_{1,d})^2}{2\sigma_{1,d}^2} d\boldsymbol{x} \\
&= -\frac{1}{2}\log(2\pi\sigma_{1,i}^2) - \frac{\mathbb{E}_q[x_i^2] - 2\mathbb{E}_q[x_i]\mu_{1,i} + \mu_{1,i}^2}{2\sigma_{1,i}^2} - \cdots - \frac{1}{2}\log(2\pi\sigma_{1,d}^2) - \frac{\mathbb{E}_q[x_d^2] - 2\mathbb{E}_q[x_d]\mu_{1,d} + \mu_{1,d}^2}{2\sigma_{1,d}^2} \\
&= -\frac{1}{2}\log(2\pi\sigma_{1,i}^2) - \frac{\sigma_{2,i}^2 + \mu_{2,i}^2 - 2\mu_{2,i}\mu_{1,i} + \mu_{1,i}^2}{2\sigma_{1,i}^2} - \cdots - \frac{1}{2}\log(2\pi\sigma_{1,d}^2) - \frac{\sigma_{2,d}^2 + \mu_{2,d}^2 - 2\mu_{2,d}\mu_{1,d} + \mu_{1,d}^2}{2\sigma_{1,d}^2} \\
&= -\frac{1}{2}\log(2\pi\sigma_{1,i}^2) - \frac{\sigma_{2,i}^2 + (\mu_{2,i}-\mu_{1,i})^2}{2\sigma_{1,i}^2} - \cdots - \frac{1}{2}\log(2\pi\sigma_{1,d}^2) - \frac{\sigma_{2,d}^2 + (\mu_{2,d}-\mu_{1,d})^2}{2\sigma_{1,d}^2} \\
&= \sum_{j}^{d} -\frac{1}{2}\log(2\pi\sigma_{1,j}^2) - \frac{\sigma_{2,j}^2}{2\sigma_{1,j}^2} - \frac{(\mu_{2,j}-\mu_{1,j})^2}{2\sigma_{1,j}^2}.
\end{aligned} \tag{B2}$$

In the following sections we derive the lower bounds presented in the main text by taking the corresponding expectations, and using Lemma 1 where it is needed. We drop the subscripts $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ from the distributions $p_\cdot(\cdot)$ and $q_\cdot(\cdot)$, respectively, to do not clutter the notation. However, we use these subscripts in the parameters $\mu_\cdot$ and $\sigma_\cdot$ to distinguish between them.

B.2 Model 1: Supervised lower bound

$$\begin{aligned}
\mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x},y)}[\log p(y)] &= \int q(\boldsymbol{z}|\boldsymbol{x},y) \log p(y) d\boldsymbol{z} \\
&= \log \boldsymbol{\pi}
\end{aligned}$$

$$\begin{aligned}
\mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x},y)}[\log p(\boldsymbol{z}|y)] &= \int q(\boldsymbol{z}|\boldsymbol{x},y) \log p(\boldsymbol{z}|y) d\boldsymbol{z} \\
&= \int \mathcal{N}(\boldsymbol{z}; \boldsymbol{\mu}_{\boldsymbol{\phi}}, \boldsymbol{\sigma}_{\boldsymbol{\phi}}^2) \log \mathcal{N}(\boldsymbol{z}; \boldsymbol{\mu}_{\boldsymbol{\theta}}, \boldsymbol{\sigma}_{\boldsymbol{\theta}}^2) d\boldsymbol{z} \\
&= -\sum_{j=1}^{\ell_z} \left(\frac{1}{2}\log(2\pi\sigma_{\boldsymbol{\theta}_{j,k}}^2) + \frac{\sigma_{\boldsymbol{\phi}_j}^2}{\sigma_{\boldsymbol{\theta}_{j,k}}^2} + \frac{(\mu_{\boldsymbol{\phi}_j} - \mu_{\boldsymbol{\theta}_{j,k}})^2}{\sigma_{\boldsymbol{\theta}_{j,k}}^2} \right)
\end{aligned}$$

$$\begin{aligned}\mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x},y)}[\log p(\boldsymbol{x}|\boldsymbol{z})] &= \int q(\boldsymbol{z}|\boldsymbol{x},y)\log p(\boldsymbol{x}|\boldsymbol{z})d\boldsymbol{z} \\ &\approx \frac{1}{L}\sum_{l=1}^{L}\log\mathcal{N}(x_i|z_{i,l})\end{aligned}$$

$$\begin{aligned}\mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x},y)}[\log q(\boldsymbol{z}|\boldsymbol{x},y)] &= \int q(\boldsymbol{z}|\boldsymbol{x},y)\log q(\boldsymbol{z}|\boldsymbol{x},y)d\boldsymbol{z} \\ &= \int \mathcal{N}(\boldsymbol{z};\boldsymbol{\mu}_{\boldsymbol{\phi}},\boldsymbol{\sigma}^2_{\boldsymbol{\phi}})\log\mathcal{N}(\boldsymbol{z};\boldsymbol{\mu}_{\boldsymbol{\phi}},\boldsymbol{\sigma}^2_{\boldsymbol{\phi}})d\boldsymbol{z} \\ &= -\sum_{j=1}^{\ell_z}(\frac{1}{2}\log(2\pi\sigma^2_{\boldsymbol{\phi}j})+1)\end{aligned}$$

B.3 Model 1: Unsupervised lower bound

$$\begin{aligned}\mathbb{E}_{q(\boldsymbol{z},y|\boldsymbol{x})}[\log p(y)] &= \sum_y \int q(y|\boldsymbol{x})q(\boldsymbol{z}|\boldsymbol{x},y)\log p(y)d\boldsymbol{z} \\ &= \log\boldsymbol{\pi}\end{aligned}$$

$$\begin{aligned}\mathbb{E}_{q(\boldsymbol{z},y|\boldsymbol{x})}[\log p(\boldsymbol{z}|y)] &= \sum_y \int q(y|\boldsymbol{x})q(\boldsymbol{z}|\boldsymbol{x},y)\log p(\boldsymbol{z}|y)d\boldsymbol{z} \\ &= \sum_y \boldsymbol{\pi}_{y|\boldsymbol{x}}\int \mathcal{N}(\boldsymbol{z};\boldsymbol{\mu}_{\boldsymbol{\phi}},\boldsymbol{\sigma}^2_{\boldsymbol{\phi}})\log\mathcal{N}(\boldsymbol{z};\boldsymbol{\mu}_{\boldsymbol{\theta}},\boldsymbol{\sigma}^2_{\boldsymbol{\theta}})d\boldsymbol{z} \\ &= -\sum_y \boldsymbol{\pi}_{y|\boldsymbol{x}}\left[\sum_{j=1}^{\ell_z}\Big(\frac{1}{2}\log(2\pi\sigma^2_{\boldsymbol{\theta}_{j,k}})+\frac{\sigma^2_{\boldsymbol{\phi}_j}}{\sigma^2_{\boldsymbol{\theta}_{j,k}}}+\frac{(\mu_{\boldsymbol{\phi}_j}-\mu_{\boldsymbol{\theta}_{j,k}})^2}{\sigma^2_{\boldsymbol{\theta}_{j,k}}}\Big)\right]\end{aligned}$$

$$\begin{aligned}\mathbb{E}_{q(\boldsymbol{z},y|\boldsymbol{x})}[\log p(\boldsymbol{x}|\boldsymbol{z})] &= \sum_y \int q(y|\boldsymbol{x})q(\boldsymbol{z}|\boldsymbol{x},y)\log p(\boldsymbol{x}|\boldsymbol{z})d\boldsymbol{z} \\ &\approx \frac{1}{L}\sum_{l=1}^{L}\log\mathcal{N}(x_i|z_{i,l})\end{aligned}$$

$$\begin{aligned}\mathbb{E}_{q(\boldsymbol{z},y|\boldsymbol{x})}[\log q(\boldsymbol{z}|\boldsymbol{x},y)] &= \sum_y \int q(y|\boldsymbol{x})q(\boldsymbol{z}|\boldsymbol{x},y)\log q(\boldsymbol{z}|\boldsymbol{x},y)d\boldsymbol{z} \\ &= \sum_y \boldsymbol{\pi}_{y|\boldsymbol{x}}\int \mathcal{N}(\boldsymbol{z};\boldsymbol{\mu}_{\boldsymbol{\phi}},\boldsymbol{\sigma}^2_{\boldsymbol{\phi}})\log\mathcal{N}(\boldsymbol{z};\boldsymbol{\mu}_{\boldsymbol{\phi}},\boldsymbol{\sigma}^2_{\boldsymbol{\phi}})d\boldsymbol{z} \\ &= -\sum_y \boldsymbol{\pi}_{y|\boldsymbol{x}}\sum_{j=1}^{\ell_z}(\frac{1}{2}\log(2\pi\sigma^2_{\boldsymbol{\phi}j})+1)\end{aligned}$$

$$\begin{aligned}\mathbb{E}_{q(\boldsymbol{z},y|\boldsymbol{x})}[\log q(y|\boldsymbol{x})] &= \sum_y \int q(y|\boldsymbol{x})q(\boldsymbol{z}|\boldsymbol{x},y)\log q(y|\boldsymbol{x})d\boldsymbol{z} \\ &= \sum_y q(y|\boldsymbol{x})\log q(y|\boldsymbol{x})\end{aligned}$$

B.4 Model 2: Supervised lower bound

$$\begin{aligned}\mathbb{E}_{q(\boldsymbol{z},\boldsymbol{a}|\boldsymbol{x},y)}[\log p(y)] &= \int\int q(\boldsymbol{a}|\boldsymbol{x})q(\boldsymbol{z}|\boldsymbol{x},y)\log p(y)d\boldsymbol{z}d\boldsymbol{a}\\ &= \log \boldsymbol{\pi}\end{aligned}$$

$$\begin{aligned}\mathbb{E}_{q(\boldsymbol{z},\boldsymbol{a}|\boldsymbol{x},y)}[\log p(\boldsymbol{z}|y)] &= \int\int q(\boldsymbol{a}|\boldsymbol{x})q(\boldsymbol{z}|\boldsymbol{x},y)\log p(\boldsymbol{z}|y)d\boldsymbol{z}d\boldsymbol{a}\\ &= \int \mathcal{N}(\boldsymbol{z};\boldsymbol{\mu}_{\boldsymbol{\phi}},\boldsymbol{\sigma}^2_{\boldsymbol{\phi}})\log\mathcal{N}(\boldsymbol{z};\boldsymbol{\mu}_{\boldsymbol{\theta}},\boldsymbol{\sigma}^2_{\boldsymbol{\theta}})d\boldsymbol{z}\\ &= -\sum_{j=1}^{\ell_z}\left(\frac{1}{2}\log(2\pi\sigma^2_{\boldsymbol{\theta}_{j,k}}) + \frac{\sigma^2_{\boldsymbol{\phi}_j}}{\sigma^2_{\boldsymbol{\theta}_{j,k}}} + \frac{(\mu_{\boldsymbol{\phi}_j}-\mu_{\boldsymbol{\theta}_{j,k}})^2}{\sigma^2_{\boldsymbol{\theta}_{j,k}}}\right)\end{aligned}$$

$$\begin{aligned}\mathbb{E}_{q(\boldsymbol{z},\boldsymbol{a}|\boldsymbol{x},y)}[\log p(\boldsymbol{x}|\boldsymbol{z},y)] &= \int\int q(\boldsymbol{a}|\boldsymbol{x})q(\boldsymbol{z}|\boldsymbol{x},y)\log p(\boldsymbol{x}|\boldsymbol{z},y)d\boldsymbol{z}d\boldsymbol{a}\\ &\approx \frac{1}{L}\sum_{l=1}^{L}\log\mathcal{N}(x_i|z_{i,l},y_i)\end{aligned}$$

$$\begin{aligned}\mathbb{E}_{q(\boldsymbol{z},\boldsymbol{a}|\boldsymbol{x},y)}[\log p(\boldsymbol{a}) - \log q(\boldsymbol{a}|\boldsymbol{x})] &= \int\int q(\boldsymbol{a}|\boldsymbol{x})q(\boldsymbol{z}|\boldsymbol{x},y)[\log p(\boldsymbol{a}) - \log q(\boldsymbol{a}|\boldsymbol{x})]d\boldsymbol{z}d\boldsymbol{a}\\ &= \int q(\boldsymbol{a}|\boldsymbol{x})\log p(\boldsymbol{a})d\boldsymbol{a} - \int q(\boldsymbol{a}|\boldsymbol{x})\log q(\boldsymbol{a}|\boldsymbol{x})d\boldsymbol{a}\\ &= -\frac{1}{2}\sum_{c=1}^{\ell_a}(\sigma^2_{\boldsymbol{\phi}_{\boldsymbol{a}_c}} + \mu^2_{\boldsymbol{\phi}_{\boldsymbol{a}_c}} - (1+\log\sigma^2_{\boldsymbol{\phi}_{\boldsymbol{a}_c}}))\end{aligned}$$

$$\begin{aligned}\mathbb{E}_{q(\boldsymbol{z},\boldsymbol{a}|\boldsymbol{x},y)}[\log q(\boldsymbol{z}|\boldsymbol{x},y)] &= \int\int q(\boldsymbol{a}|\boldsymbol{x})q(\boldsymbol{z}|\boldsymbol{x},y)\log q(\boldsymbol{z}|\boldsymbol{x},y)d\boldsymbol{z}d\boldsymbol{a}\\ &= \int q(\boldsymbol{z}|\boldsymbol{x},y)\log q(\boldsymbol{z}|\boldsymbol{x},y)d\boldsymbol{z}\\ &= \frac{1}{2}\sum_{j=1}^{\ell_z}(1+\log\sigma^2_{\boldsymbol{\phi}_{\boldsymbol{z}_j}})\end{aligned}$$

B.5 Model 2: Unsupervised lower bound

$$\begin{aligned}\mathbb{E}_{q(\boldsymbol{z},\boldsymbol{a},y|\boldsymbol{x})}[\log p(y)] &= \int\sum_y\int q(\boldsymbol{a}|\boldsymbol{x})q(y|\boldsymbol{x},\boldsymbol{a})q(\boldsymbol{z}|\boldsymbol{x},y)\log p(y)d\boldsymbol{z}d\boldsymbol{a}\\ &= \log\boldsymbol{\pi}\end{aligned}$$

$$\begin{aligned}\mathbb{E}_{q(\boldsymbol{z},\boldsymbol{a},y|\boldsymbol{x})}[\log q(y|\boldsymbol{x},\boldsymbol{a})] &= \int\sum_y\int q(\boldsymbol{a}|\boldsymbol{x})q(y|\boldsymbol{x},\boldsymbol{a})q(\boldsymbol{z}|\boldsymbol{x},y)\log q(y|\boldsymbol{x},\boldsymbol{a})d\boldsymbol{z}d\boldsymbol{a}\\ &\approx \frac{1}{L_a}\sum_{l_a=1}^{L_a}\sum_y q(y|\boldsymbol{x},\boldsymbol{a}_{l_a})\log q(y|\boldsymbol{x},\boldsymbol{a}_{l_a})\end{aligned}$$

$$\begin{aligned}
\mathbb{E}_{q(\boldsymbol{z},\boldsymbol{a},y|\boldsymbol{x})}[\log p(\boldsymbol{z}|y)] &= \int \sum_y \int q(\boldsymbol{a}|\boldsymbol{x}) q(y|\boldsymbol{x},\boldsymbol{a}) q(\boldsymbol{z}|\boldsymbol{x},y) \log p(\boldsymbol{z}|y) d\boldsymbol{z} d\boldsymbol{a} \\
&\approx \frac{1}{L_a} \sum_{l_a=1}^{L_a} \sum_y q(y|\boldsymbol{x},\boldsymbol{a}_{l_a}) \int q(\boldsymbol{z}|\boldsymbol{x},y_{l_a}) \log p(\boldsymbol{z}|y_{l_a}) d\boldsymbol{z} \\
&\approx -\frac{1}{L_a} \sum_{l_a=1}^{L_a} \sum_y \boldsymbol{\pi}_{y|\boldsymbol{x},a_{l_a}} \Big[\sum_{j=1}^{\ell_z} \Big(\frac{1}{2} \log(2\pi\sigma^2_{\boldsymbol{\theta}_{j,k}}) + \frac{\sigma^2_{\boldsymbol{\phi}_j}}{\sigma^2_{\boldsymbol{\theta}_{j,k}}} \\
&+ \frac{(\mu_{\boldsymbol{\phi}_j} - \mu_{\boldsymbol{\theta}_{j,k}})^2}{\sigma^2_{\theta_{j,k}}} \Big) \Big]
\end{aligned}$$

$$\begin{aligned}
\mathbb{E}_{q(\boldsymbol{z},\boldsymbol{a},y|\boldsymbol{x})}[\log p(\boldsymbol{x}|\boldsymbol{z},y)] &= \int \sum_y \int q(\boldsymbol{a}|\boldsymbol{x}) q(y|\boldsymbol{x},\boldsymbol{a}) q(\boldsymbol{z}|\boldsymbol{x},y) \log p(\boldsymbol{x}|\boldsymbol{z},y) d\boldsymbol{z} d\boldsymbol{a} \\
&\approx \frac{1}{L_a} \sum_{l_a=1}^{L_a} \sum_y \boldsymbol{\pi}_{y|\boldsymbol{x},\boldsymbol{a}_{l_a}} \frac{1}{L_z} \sum_{l_z=1}^{L_z} \log \mathcal{N}(\boldsymbol{x}_i|\boldsymbol{z}_{i,l}, y_{l_a})
\end{aligned}$$

$$\begin{aligned}
\mathbb{E}_{q(\boldsymbol{z},\boldsymbol{a},y|\boldsymbol{x})}[\log p(\boldsymbol{a}) - \log q(\boldsymbol{a}|\boldsymbol{x})] &= \int \sum_y \int q(\boldsymbol{a}|\boldsymbol{x}) q(y|\boldsymbol{x},\boldsymbol{a}) q(\boldsymbol{z}|\boldsymbol{x},y) [\log p(\boldsymbol{a}) - \log q(\boldsymbol{a}|\boldsymbol{x})] d\boldsymbol{z} d\boldsymbol{a} \\
&= \sum_y q(\boldsymbol{a}|\boldsymbol{x}) \Big[\int q(y|\boldsymbol{x},\boldsymbol{a}) \log p(\boldsymbol{a}) d\boldsymbol{a} - \int q(\boldsymbol{a}|\boldsymbol{x}) \log q(\boldsymbol{a}|\boldsymbol{x}) d\boldsymbol{a} \Big] \\
&= -\frac{1}{2} \sum_y \boldsymbol{\pi}_{y|\boldsymbol{x},a_{l_a}} \Big[\sum_{c=1}^{\ell_a} (\sigma^2_{\boldsymbol{\phi}_{\boldsymbol{a}_c}} + \mu^2_{\boldsymbol{\phi}_{\boldsymbol{a}_c}} - (1 + \log \sigma^2_{\boldsymbol{\phi}_{\boldsymbol{a}_c}})) \Big]
\end{aligned}$$

$$\begin{aligned}
\mathbb{E}_{q(\boldsymbol{z},\boldsymbol{a},y|\boldsymbol{x})}[\log q(\boldsymbol{z}|\boldsymbol{x},y)] &= \int \sum_y \int q(\boldsymbol{a}|\boldsymbol{x}) q(y|\boldsymbol{x},\boldsymbol{a}) q(\boldsymbol{z}|\boldsymbol{x},y) \log q(\boldsymbol{z}|\boldsymbol{x},y) d\boldsymbol{z} d\boldsymbol{a} \\
&\approx \frac{1}{L_a} \sum_{l_a=1}^{L_a} \sum_y q(y|\boldsymbol{x},\boldsymbol{a}_{l_a}) \int q(\boldsymbol{z}|\boldsymbol{x},y) \log q(\boldsymbol{z}|\boldsymbol{x},y) d\boldsymbol{z} \\
&= -\frac{1}{L_a} \sum_{l_a=1}^{L_a} \sum_y \boldsymbol{\pi}_{y,\boldsymbol{a}_{l_a}} \Big[\frac{1}{2} \sum_{j=1}^{\ell_z} (1 + \log \sigma^2_{\boldsymbol{\phi}_{\boldsymbol{z}_j}}) \Big]
\end{aligned}$$

References

[1] Raymond Anderson. *The credit scoring toolkit: theory and practice for retail credit risk management and decision automation*. Oxford University Press, 2007.

[2] Michael Bücker, Maarten van Kampen, and Walter Krämer. Reject inference in consumer credit scoring with nonignorable missing data. *Journal of Banking & Finance*, 37(3):1040–1045, 2013.

[3] Ha-Thu Nguyen. Reject inference in application scorecards: evidence from France. EconomiX Working Papers 2016-10, University of Paris Nanterre, EconomiX, 2016. URL `https://ideas.repec.org/p/drm/wpaper/2016-10.html`.

[4] G Gary Chen and Thomas Astebro. The economic value of reject inference in credit scoring. *Department of Management Science, University of Waterloo*, 2001.

[5] Andrew Marshall, Leilei Tang, and Alistair Milne. Variable reduction, sample selection bias and bank retail credit scoring. *Journal of Empirical Finance*, 17(3):501–512, 2010.

[6] David J Hand and William E Henley. Can reject inference ever work? *IMA Journal of Management Mathematics*, 5(1):45–55, 1993.

[7] Diederik P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems*, pages 3581–3589, 2014.

[8] Lars Maaløe, Casper Kaae Sønderby, Søren Kaae Sønderby, and Ole Winther. Auxiliary deep generative models. *arXiv preprint arXiv:1602.05473*, 2016.

[9] Chuhan Wu, Fangzhao Wu, Sixing Wu, Zhigang Yuan, Junxin Liu, and Yongfeng Huang. Semi-supervised dimensional sentiment analysis with variational autoencoder. *Knowledge-Based Systems*, 165:30–39, 2019.

[10] Xianghua Fu, Yanzhi Wei, Fan Xu, Ting Wang, Yu Lu, Jianqiang Li, and Joshua Zhexue Huang. Semi-supervised aspect-level sentiment classification model based on variational autoencoder. *Knowledge-Based Systems*, 171:81–92, 2019.

[11] Yin Zheng, Huachun Tan, Bangsheng Tang, Hanning Zhou, et al. Variational deep embedding: A generative approach to clustering. arxiv preprint. *arXiv preprint arXiv:1611.05148*, 1(2):5, 2016.

[12] Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349*, 2015.

[13] Xianxu Hou, Linlin Shen, Ke Sun, and Guoping Qiu. Deep feature consistent variational autoencoder. In *Applications of Computer Vision (WACV), 2017 IEEE Winter Conference on*, pages 1133–1141. IEEE, 2017.

[14] Siddique Latif, Rajib Rana, Junaid Qadir, and Julien Epps. Variational autoencoders for learning latent representations of speech emotion. *arXiv preprint arXiv:1712.08708*, 2017.

[15] Rogelio Andrade Mancisidor, Michael Kampffmeyer, Kjersti Aas, and Robert Jenssen. Learning latent representations of bank customers with the variational autoencoder. *arXiv preprint arXiv:1903.06580*, 2019.

[16] Lars Maaløe, Casper Kaae Sønderby, Søren Kaae Sønderby, and Ole Winther. Improving semi-supervised learning with auxiliary deep generative models. In *NIPS Workshop on Advances in Approximate Bayesian Inference*, 2015.

[17] David C Hsia. Credit scoring and the equal credit opportunity act. *Hastings LJ*, 30:371, 1978.

[18] C. Rosenberg, M. Hebert, and H. Schneiderman. Semi-supervised self-training of object detection models. In *2005 Seventh IEEE Workshops on Applications of Computer Vision*, volume 1, pages 29–36, Jan 2005. doi: 10.1109/ACVMOT.2005.107.

[19] Fabian Gieseke, Antti Airola, Tapio Pahikkala, and Oliver Kramer. Sparse quasi-newton optimization for semi-supervised support vector machines. In *ICPRAM (1)*, pages 45–54, 2012.

[20] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[21] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[22] Lyn C Thomas. A survey of credit and behavioural scoring: forecasting financial risk of lending to consumers. *International journal of forecasting*, 16(2):149–172, 2000.

[23] Zhiyong Li, Ye Tian, Ke Li, Fanyin Zhou, and Wei Yang. Reject inference in credit scoring using semi-supervised support vector machines. *Expert Systems with Applications*, 74:105–114, 2017.

[24] AJ Feelders. Credit scoring and reject inference with mixture models. *Intelligent Systems in Accounting, Finance & Management*, 9(1):1–8, 2000.

[25] Derrick N Joanes. Reject inference applied to logistic regression for credit scoring. *IMA Journal of Management Mathematics*, 5(1):35–43, 1993.

[26] Jonathan Banasik, John Crook, and Lyn Thomas. Sample selection bias in credit scoring models. *Journal of the Operational Research Society*, 54(8):822–832, 2003.

[27] Jonathan Crook and John Banasik. Does reject inference really improve the performance of application scoring models? *Journal of Banking & Finance*, 28(4):857–874, 2004.

[28] Geert Verstraeten and Dirk Van den Poel. The impact of sample bias on consumer credit scoring performance and profitability. *Journal of the operational research society*, 56(8):981–992, 2005.

[29] J Banasik and J Crook. Credit scoring, augmentation and lean models. *Journal of the Operational Research Society*, 56(9):1072–1081, 2005. doi: 10.1057/palgrave.jors.2602017.

[30] So Young Sohn and HW Shin. Reject inference in credit operations based on survival analysis. *Expert Systems with Applications*, 31(1):26–29, 2006.

[31] John Banasik and Jonathan Crook. Reject inference, augmentation, and sample selection. *European Journal of Operational Research*, 183(3):1582–1594, 2007.

[32] Y Kim and SY Sohn. Technology scoring model considering rejected applicants and effect of reject inference. *Journal of the Operational Research Society*, 58(10):1341–1347, 2007.

[33] I-Ding Wu and David J Hand. Handling selection bias when choosing actions in retail credit applications. *European journal of operational research*, 183(3):1560–1568, 2007.

[34] J. Banasik and J. Crook. Reject inference in survival analysis by augmentation. *The Journal of the Operational Research Society*, 61(3):473–485, 2010.

[35] Sebastián Maldonado and Gonzalo Paredes. A semi-supervised approach for reject inference in credit scoring using svms. In *Industrial Conference on Data Mining*, pages 558–571. Springer, 2010.

[36] Gongyue Gary Chen and Thomas Åstebro. Bound and collapse bayesian reject inference for credit scoring. *Journal of the Operational Research Society*, 63(10):1374–1387, 2012.

[37] Billie Anderson and J Michael Hardin. Modified logistic regression using the em algorithm for reject inference. *International Journal of Data Analysis Techniques and Strategies*, 5(4):359–373, 2013.

[38] Dennis Ash and Steve Meester. Best practices in reject inferencing. *Conference on Credit Risk Modeling and Decisioning: Philadelphia, PA.*, 01 2002.

[39] James J Heckman. The common structure of statistical models of truncation, sample selection and limited dependent variables and a simple estimator for such models. In *Annals of Economic and Social Measurement, Volume 5, number 4*, pages 475–492. NBER, 1976.

[40] James J. Heckman. Sample selection bias as a specification error. *Econometrica*, 47(1):153–161, 1979.

[41] William J Boyes, Dennis L Hoffman, and Stuart A Low. An econometric analysis of the bank credit scoring problem. *Journal of Econometrics*, 40(1):3–14, 1989.

[42] William Greene. Sample selection in credit-scoring models1. *Japan and the world Economy*, 10(3): 299–316, 1998.

[43] Patrick Puhani. The heckman correction for sample selection and its critique. *Journal of economic surveys*, 14(1):53–68, 2000.

[44] Ye Tian and Jian Luo. A new branch-and-bound approach to semi-supervised support vector machine. *Soft Comput.*, 21(1):245–254, 2017. doi: 10.1007/s00500-016-2089-y. URL `https://doi.org/10.1007/s00500-016-2089-y`.

[45] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[46] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

[47] Håvard Kvamme, Nikolai Sellereite, Kjersti Aas, and Steffen Sjursen. Predicting mortgage default using convolutional neural networks. *Expert Systems with Applications*, 102:207–217, 2018.

[48] Cheng Zhang, Judith Butepage, Hedvig Kjellstrom, and Stephan Mandt. Advances in variational inference. *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[49] Meelis Kull, Telmo Silva Filho, and Peter Flach. Beta calibration: a well-founded and easily implemented improvement on logistic calibration for binary classifiers. In *Artificial Intelligence and Statistics*, pages 623–631, 2017.

[50] Theano Development Team. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688, May 2016. URL `http://arxiv.org/abs/1605.02688`.

[51] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[52] Mengchen Liu, Jiaxin Shi, Kelei Cao, Jun Zhu, and Shixia Liu. Analyzing the training processes of deep generative models. *IEEE transactions on visualization and computer graphics*, 24(1):77–87, 2018.

[53] Hiroshi Takahashi, Tomoharu Iwata, Yuki Yamanaka, Masanori Yamada, and Satoshi Yagi. Student-t variational autoencoder for robust density estimation. In *IJCAI*, pages 2696–2702, 2018.

[54] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010.

[55] John Platt et al. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74, 1999.